Extracting Personal Information from Conversations

A dissertation submitted towards the degree
Doctor of Engineering (Dr.-Ing.)
of the Faculty of Mathematics and Computer Science
of Saarland University

by
Anna Tigunova

Saarbrücken
2021

Defense Colloquium

Date: 18 February 2022

Dean of the Faculty: Prof. Dr. Thomas Schuster

Examination Committee

Chair: Prof. Anna Maria Feit

Reviewer, Advisor: Prof. Dr. Gerhard Weikum

Reviewer, Co-Advisor: Dr. Andrew Yates

Reviewer: Prof. Dr. Vera Demberg

Academic Assistant: Dr. Vladislav Golyanik

Acknowledgements

I would like to thank my advisor, Professor Gerhard Weikum and my collaborators, Paramita Mirza and Andrew Yates for their supervision and encouragement.

I want to thank my friends, Azin Zahraei, Konstantin Poddubnyy, Marin Schmädeke, Kevin Jose and Harpreet Singh, for being there for me.

Also many thanks to my colleagues at D5; my parents and bff Alexandra and Marya.

Abstract

PERSONAL knowledge is a versatile resource that is valuable for a wide range of downstream applications. Background facts about users can allow chatbot assistants to produce more topical and empathic replies. In the context of recommendation and retrieval models, personal facts can be used to customize the ranking results for individual users.

A *Personal Knowledge Base*, populated with personal facts, such as demographic information, interests and interpersonal relationships, is a unique endpoint for storing and querying personal knowledge. Such knowledge bases are easily interpretable and can provide users with full control over their own personal knowledge, including revising stored facts and managing access by downstream services for personalization purposes.

To alleviate users from extensive manual effort to build such personal knowledge base, we can leverage automated extraction methods applied to the textual content of the users, such as dialogue transcripts or social media posts. Mainstream extraction methods specialize on well-structured data, such as biographical texts or encyclopedic articles, which are rare for most people. In turn, conversational data is abundant but challenging to process and requires specialized methods for extraction of personal facts.

In this dissertation we address the acquisition of personal knowledge from conversational data. We propose several novel deep learning models for inferring speakers' personal attributes:

- Demographic attributes, *age*, *gender*, *profession* and *family status*, are inferred by **HAMs** - hierarchical neural classifiers with attention mechanism. Trained HAMs can be transferred between different types of conversational data and provide interpretable predictions.

- Long-tailed personal attributes, *hobby* and *profession*, are predicted with **CHARM** - a zero-shot learning model, overcoming the lack of labeled training samples for rare attribute values. By linking conversational utterances to external sources, CHARM is able to predict attribute values which it never saw during training.

- *Interpersonal relationships* are inferred with **PRIDE** - a hierarchical transformer-based model. To accurately predict fine-grained relationships, PRIDE leverages personal traits of the speakers and the style of conversational utterances.

Experiments with various conversational texts, including Reddit discussions and movie scripts, demonstrate the viability of our methods and their superior performance compared to state-of-the-art baselines.

Kurzfassung

Personengebundene Fakten sind eine vielseitig nutzbare Quelle für die verschiedensten Anwendungen. Hintergrundfakten über Nutzer können es Chatbot-Assistenten ermöglichen, relevantere und persönlichere Antworten zu geben. Im Kontext von Empfehlungs- und Retrievalmodellen können personengebundene Fakten dazu verwendet werden, die Ranking-Ergebnisse für Nutzer individuell anzupassen.

Eine *Personengebundene Wissensdatenbank*, gefüllt mit persönlichen Daten wie demografischen Angaben, Interessen und Beziehungen, kann eine universelle Schnittstelle für die Speicherung und Abfrage solcher Fakten sein. Wissensdatenbanken sind leicht zu interpretieren und bieten dem Nutzer die vollständige Kontrolle über seine personenbezogenen Fakten, einschließlich der Überarbeitung und der Verwaltung des Zugriffs durch nachgelagerte Dienste, etwa für Personalisierungszwecke.

Um den Nutzern den aufwändigen manuellen Aufbau einer solchen persönlichen Wissensdatenbank zu ersparen, können automatisierte Extraktionsmethoden auf den textuellen Inhalten der Nutzer – wie z.B. Konversationen oder Beiträge in sozialen Medien – angewendet werden. Die üblichen Extraktionsmethoden sind auf strukturierte Daten wie biografische Texte oder enzyklopädische Artikel spezialisiert, die bei den meisten Menschen keine Rolle spielen.

In dieser Dissertation beschäftigen wir uns mit der Gewinnung von persönlichem Wissen aus Dialogdaten und schlagen mehrere neuartige Deep-Learning-Modelle zur Ableitung persönlicher Attribute von Sprechern vor:

- Demographische Attribute wie *Alter*, *Geschlecht*, *Beruf* und *Familienstand* werden durch **HAMs** - Hierarchische Neuronale Klassifikatoren mit Attention-Mechanismus - abgeleitet. Trainierte HAMs können zwischen verschiedenen Arten von Gesprächsdaten übertragen werden und liefern interpretierbare Vorhersagen

- Vielseitige persönliche Attribute wie *Hobbys* oder *Beruf* werden mit **CHARM** ermittelt - einem Zero-Shot-Lernmodell, das den Mangel an markierten Trainingsbeispielen für seltene Attributwerte überwindet. Durch die Verknüpfung von Gesprächsäußerungen mit externen Quellen ist CHARM in der Lage, Attributwerte zu ermitteln, die es beim Training nie gesehen hat

- *Zwischenmenschliche Beziehungen* werden mit **PRIDE**, einem hierarchischen transformerbasierten Modell, abgeleitet. Um präzise Beziehungen vorhersagen zu können, nutzt PRIDE persönliche Eigenschaften der Sprecher und den Stil von Konversationsäußerungen

Experimente mit verschiedenen Konversationstexten, inklusive Reddit-Diskussionen und Filmskripten, demonstrieren die Praxistauglichkeit unserer Methoden und ihre hervorragende Leistung im Vergleich zum aktuellen Stand der Technik.

Contents

Chapter 1

Introduction

Contents

1.1 Motivation and Problem Statement

The recent rise of social media enabled the access to vast amounts of user-generated content. This data has many potential applications; *personalization* being a major use case.

Having background information about the user is practical for many downstream applications, such as recommender systems and search engines [7]. For instance, the ranking results for the query "*best weekend activities in Vancouver*" can be rearranged based on the user's known hobbies to deliver more topical content for a particular individual. Another application greatly benefiting from the availability of the user's background is personalized intelligent assistants.

Chat-bots have become an essential part of everyday life, being able to deal with a wide range of routine tasks, provide factual information and entertain the user. Still, there is a growing demand for creating user-oriented intelligent agents, which are able to hold personalized conversations, while at the same time building and expanding their knowledge repository about the user's traits and preferences.

Having the access to the user's interests and background, personalized chat-bots will be able to build relevant responses and start new topics, interesting for the user. The ability to produce meaningful or even funny and surprising utterances is a desired feature, making a chat-bot appealing to the user. Therefore, there has been significant research interest in personalizing intelligent assistants, which still remains challenging.

Consider, for example, the following conversation between a human (H) and an intelligent assistant (A), illustrating the need of the intelligent assistant to have background information about its interlocutor:

H: *Can you text my wife I will be late from the hospital? Need to do an urgent operation.*
A: *Sure.*
H: *Any idea where I can get dinner that late?*
A: *I'd suggest Ocean Thai Cafe, they are open till midnight.*
H: *Sounds good! Talking about oceans, can you remind me to pack my goggles? Last time I went to the pool I forgot them.*

To effectively address the user's requests from the conversation above, the chat-bot has to know the following facts about the user: *(i) interpersonal relationships* (knowing who the user's *wife* is, to be able to contact her) and *(ii) food preferences* (suggest an appropriate restaurant, knowing that the user loves *thai* food).

To alleviate users from extensive manual effort to provide their information, the intelligent assistant should be capable of learning it directly from conversational utterances and digital traces of the user. For example, from the dialogue above the chat-bot can infer the user's profession *surgeon* using the cues "*hospital*" and "*operation*" and the user's hobby *swimming* from the words "*goggles*" and "*pool*". Such automatic extraction is an extremely challenging but interesting and useful task.

The work in this dissertation concerns predicting personal facts from textual representation of conversations. The sources of such data include transcribed everyday dialogues, social media submissions or chats in messenger apps. Information extraction from conversational data is a challenging task [170, 176], which is still underexplored in related studies.

1.2 Personal Knowledge Base

The format in which the personal information will be extracted, stored and used is an important issue to consider. Concerning this, we propose constructing a *Personal Knowledge Base* (PKB) [7, 176], a structured source of information about a particular individual. Such information could be the user's demographic facts, her interests, relationships or personal possessions.

Design and construction of PKBs has recently gained significant research interest [7, 156]. While some studies propose creating a PKB as a lifelog, consisting of a collection of life events [64, 176], our view of a PKB is analogous to the definition of a *Personal Knowledge Graph* given by Balog and Kenter [7], to be a structured collection of entities personally related to the user.

We outline several properties of a PKB, which make it desirable to use in various applications:

- *Transferable:* PKB acts as a unique endpoint for storing and querying user information for a wide range of applications, regardless of their internal data representation.
- *Well-structured:* the information in a PKB should be stored in a consolidated and uniform format, for example, in the form of triplets $\langle user, attribute, value \rangle$ (e.g., $\langle user1, hobby, diving \rangle$), which makes searching for specific personal information easier.

- *Owned by the user:* the users are provided with full control over their PKB, such as viewing and editing the stored information and managing access by various services.
- *Interpretable:* explicit format of the personal facts in a PKB allows the users to have the explanations about any personalized decisions made by the applications accessing their PKB.

As noted in Gerritse et al. [43] PKBs can be involved in introducing bias based on specific personal traits. Indeed, while being a tool to enhance user experience and deliver relevant content, the information in a PKB, such as *gender* or *ethnicity* of the user, can lead to discrimination when used by personalized applications [1, 29]. In this light, enabling the users to hide any sensitive facts in their PKB is a highly valuable feature.

One example of such storage of users' personal facts is Google Ads Settings[1]. This service has some features of a PKB, such as storing explicit facts and allowing the users to modify them. However, the information in Ads Settings is not structured; for example, the preferences of the user are kept as a list of concepts, not separated into fine-grained categories (like *favorite food* or *preferred travel destination*). Also the information in Ads Settings is controlled by Google - collected from Google services and used for advertising within them; therefore, the user can not restrict the access for any particular applications.

1.2.0.1 Personal Knowledge Base from Conversational Data

As mentioned in the previous section, a PKB can be populated by automatically extracting personal facts from users' conversational data. Mining personal knowledge from user-generated content to populate PKBs, or *user profiling*, is a long-standing topic in Natural Language Processing [9, 40].

Creation of a PKB from conversational data requires addressing the following key issues:

- Which personal facts are relevant and feasible to extract?
- How can this knowledge be inferred from conversational utterances?
- What are the potential applications of the data?

Concerning the first issue, we define personal facts to be user's demographic attributes (age, gender, origin, etc), hobbies and interests, interpersonal relationships (family status, names of friends, etc), skills, personality values or sentiments towards people and specific topics. Many of those attributes are subjective or mutable, making them specifically challenging to extract and process.

The second issue is concerned with information extraction from text. Prior works have mostly focused on well-comprehensible text genres, such as Wikipedia articles or news stories; however, such methods do not work as well given conversations as input. Compared to formal documents, dialogues are noisy, utterances are short [11], the language used is colloquial and the topics are diverse (including smalltalk). The dialogue utterances often give merely implicit cues about the speakers, making well established pattern-based extraction methods inapplicable.

[1] https://adssettings.google.com

Given that the personal information in conversational utterances is rarely stated explicitly, many prior studies opt for creating *latent* user representations. We argue, however, that creating an explicit PKB containing distilled personal facts provides the following advantages:

- The knowledge in a PKB can easily be shared among multiple applications, as it is not bound to some latent representation produced by a specific model. The stored facts can be reused and updated by any application, supporting the scenario of multiple repeated interactions with the user.
- A PKB is transparent and interpretable for the end users, providing them with full control over their personal knowledge, including revising stored facts and managing access of the downstream services for personalization purposes.

A detailed exploration of the third issue is beyond the scope of this thesis. We identify the potential applications of explicit personal facts to be personalized recommender systems, news feeds and content suggestions (for example, in video streaming services). Moreover, personal information can enhance the ranking results of search engines. Lastly, as motivated by the example in the previous section, the personal background knowledge can be used to produce topically focused and user-friendly responses by intelligent assistants.

1.3 Task Formulation

In our research we focus on inferring personal facts from conversational data. We work with crisp personal attributes, such as *profession* or *age*, ensuring that for all explored attributes we can define a finite list of possible values. In this work we do not consider subjective, changeable (*sentiments*) and open-ended (*favorite song*) attributes. We investigate personal attribute extraction from user-generated textual conversational data, such as transcribed spoken dialogues and social media submissions.

1.4 State of the Art and its Limitations

The related studies generally utilize pattern-based approaches, searching for explicit mentions of personal attributes in speakers' utterances, such as extracting *profession*: *software engineer* and *employment_history*: *Microsoft* from *"I work for Microsoft as a software engineer"* [85]. Such methods are limited by their inability to consider implicit contexts (e.g., *"I write product code in Redmond."*), and routinely perform worse than methods based on inference.

Most inference-based methods predict personal attributes with a small set of values, often arranged in coarse-grained categories (e.g. predicting *occupational class* instead of a fine-grained profession) or even modeled as a binary task (*age*: *young/old* [87], *political orientation*: *democrat/republican* [124]). The inference of long-tailed personal attributes with a large number of values (like *hobby* or *favorite food type*), has mostly been overlooked in previous work.

On the other hand, there has been significant research effort on creating latent representations of speakers [83, 88]. Such representations can be directly used for response generation

in a dialogue system [186]. Yet, such information is not scrutable, providing no possibility for the speaker to view and change it. Additionally, latent representations are difficult to transfer between models and applications.

1.5 Contributions

Within this dissertation we develop novel approaches for inferring personal knowledge from conversations, which address the limitations of the prior work. The contributions of this thesis can be summarized as follows. We introduce neural learning models tailored specifically for prediction of personal attributes:

- **HAM**, a light-weight model for inferring demographic facts: *gender*, *age*, *occupation* and *family status*. HAM is based on attention mechanisms, allowing to inspect and interpret its predictions.
- **CHARM**, a zero-shot learning model for predicting long-tailed personal attributes, *profession* and *hobby*. CHARM can predict rare attribute values (such as *hobby:curling*) without having training data for them.
- **PRIDE**, a transformer-based model for predicting directed fine-grained *interpersonal relationships* of the speakers in dyadic conversations.

To support our experiments we create and release large-scale conversational datasets, labeled with personal attributes. Our datasets come in two flavours: conversational transcripts (movie and series scripts) and social media submissions (Reddit discussion threads). To the best of our knowledge, our datasets are the biggest and most comprehensive collections containing conversational data with personal attribute labels.

Finally, we conduct extensive experiments to show the viability of our models and their superior performance compared to the state-of-the-art baselines. We inspect the interpretability of the developed methods and perform stress tests in the transfer learning setup.

In summary, we provide a list of publications, from which this dissertation includes material:

- Tigunova, A., Yates, A., Mirza, P., Weikum, G. (2019, May). **Listening between the lines: Learning personal attributes from conversations.** *In The World Wide Web Conference* (pp. 1818-1828).
- Tigunova, A., Mirza, P., Yates, A., Weikum, G. (2020, May). **RedDust: a Large Reusable Dataset of Reddit User Traits.** *In Proceedings of the 12th Language Resources and Evaluation Conference* (pp. 6118-6126).
- Tigunova, A., Yates, A., Mirza, P., Weikum, G. (2020, November). **CHARM: Inferring Personal Attributes from Conversations.** *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (pp. 5391-5404).

- Tigunova, A., Mirza, P., Yates, A., Weikum, G. (2021, March). **Exploring Personal Knowledge Extraction from Conversations with CHARM.** *In Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (pp. 1077-1080).
- Tigunova, A., Mirza, P., Yates, A., Weikum, G. (2021, November). **PRIDE: Predicting Relationships from Conversations.** *In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (to appear).

Additionally the contents of Chapters 4 and 5 of this dissertation were presented at Doctoral Consortium of the Web Conference 2020[2].

1.6 Organisation

The rest of this thesis is organized as follows. In Chapter 2 we discuss relevant prior studies and give the necessary methodological background. Chapter 3 presents the conversational datasets we created to support the experiments in our work. In Chapters 4, 5 and 6 we describe our novel models for inferring personal attributes: HAM, CHARM and PRIDE, respectively. Finally, we conclude the dissertation in Chapter 7 and outline possible directions for future research.

[2]Anna Tigunova. 2020. **Extracting Personal Information from Conversations.** In Companion Proceedings of the Web Conference 2020 (pp. 284–288).

Chapter 2

Background

Contents

In this chapter we cover the necessary background for our work on predicting personal information from conversations. In Section 2.1 we discuss previous studies, giving a brief overview of the methods for inferring personal attributes and interpersonal relationships in transcribed dialogues and social media submissions. In Section 2.2 we provide theoretical details for the approaches used in our work: background on deep learning for natural language processing and model evaluation criteria.

2.1 Related work

This section discusses prior work on methods to extract and represent personal knowledge in conversations. Our research concerns the textual representation of conversations, i.e. transcribed dialogues. We distinguish two general types of attributes, which can be extracted from conversations to populate a personal knowledge base: *personal attributes* and *interpersonal relationships*.

2.1.1 Personal attributes

Textual sources of dialogue data range from conversations between people (e.g. transcribed phone dialogues [42, 66]) to user-chatbot interactions, which could be open-domain [83, 183] or task-oriented [61, 96, 126]. Another distinguishable dialogue source is literary plays and

film scripts [59, 88, 110]. Other direction of related work utilizes conversational data from online sources, such as social media platforms [116, 123, 138], emails [42] and blogs [138].

2.1.1.1 Demographic attributes from transcribed dialogues

Prior work uses conversations (such as messages or telephone interactions) to extract speaker's demographic facts, which vary from gender [42, 85, 165], age [42, 165] or ethnicity [42, 85] to biography facts [60].

Most prior work constructs a speaker's latent representation using linguistic feature sets [42, 60, 88], language models [147] or embeddings [83, 96]. The disadvantage of creating such latent personality is its limited interpretability, preventing from checking its consistency with explicit attributes.

A personal profile can also be viewed as a textual description [183], predicted from a pool of candidate sentences. This representation, however, provides no exact facts, which can be readily inserted into a personal knowledge base.

Few research efforts are dedicated to distilling precise personal facts [42, 85, 165], often requiring the search for explicit pattern mentions ("*I am a doctor*") [85]. However, such assertions are rare in real conversations, yielding a poor recall of the models. Instead, some authors resort to classification [42, 88] via linguistic features. Additionally, Garera and Yarowsky incorporate partner identity and n-grams to classify age or gender using Support Vector Machines (SVMs) [42]. A major drawback of their work is that the inferred attributes take only binary values.

Wu et al. [170] infer personal attributes in the form of $\langle$*subject*, *predicate*, *object*$\rangle$ triplets, where subject and object are generated over all vocabulary terms. This makes the list of possible predicted values open-ended, as opposed to the approach of Li et al. [85], requiring the values to be present in the text. However, Wu et al. [170] generate predictions on per-utterance basis, ignoring the repeated evidence from the full history of user's conversations, which prevents from efficiently building a personal knowledge base.

Wang et al. [164] create $\langle$*subject*, *predicate*, *object*$\rangle$ triplets by generating predicate and object, assuming that the subject is always the speaker. The major limitation of this work is that the generation is constrained by the values present in the training data, which naturally limits the applicability of the proposed approach to slightly different conversational datasets. Wang et al. [164] also make predictions on each separate sentence, making a strong assumption that each input sentence contains some personal attribute information.

To predict explicit or latent personality, standard machine learning tools are commonly used, such as Logistic Regression [3] or Conditional Random Fields [83], which often operate on hand-crafted linguistic features [42]. Some works exploit conversational partner information, including their identity [3], personal traits [42] or their utterances [165]. More recently, the speaker's representations are built using neural approaches [61, 96, 165, 183].

2.1.1.2 Social media profiling

The texts stemming from social media are often colloquial, noisy and short [11], making them similar to spoken utterances. Moreover, the discussion threads in topical forums

resemble the flow of a natural conversation. Most of existing work on user profiling focuses on Twitter [65, 70, 83, 116, 123, 130, 138, 176]. Other research explores Facebook [100, 138], Reddit [46] and other platforms [186]. Nevertheless, social media content is still different from natural conversational text, due to the additional signals (user activity or hashtags), which are often used alongside the texts [46, 130].

There has been significant effort on predicting age and gender of social media users [70, 86, 130, 138, 186] and their personality [46, 100]. Less explored but more specific attributes like origin [130], political views [116, 130], ethnicity [116], occupation [82, 123], education [82] or mental health issues [145] are also considered.

The representations utilized for social media profiling are built via linguistic features [46, 100], language models [138], n-grams [130] or embeddings [123]. These features are used afterwards in SVMs [46, 130], topical models [116] and neural networks [70, 186].

A common limitation of most prior work is the small number of attribute values to be predicted. In contrast, Li et al. [82] predicts education, job and spouses of the users without exploiting predefined attribute value lists, as the prediction is based in the entities found in user's posts. This approach relies on detecting explicit mentions, yet it relaxes the restriction to know all attribute values in advance. Another example of an open-ended personal attribute is user's interests, which has also received significant research interest.

2.1.1.3 User interests from social media

Extracting the users' topics of interest (e.g., *music*, *politics* or *sports*) has received significant attention, as they are highly helpful for recommendations of news or publications. As opposed to disclosing personal facts, speakers are more willingly talking about what they are interested in [128, 142], making extraction of interests from conversational utterances more feasible. However, the interests are often changeable [119], requiring revisions to the past predictions.

Several studies discover speakers' interests using supervised learning techniques [128], training machine learning models to predict the interests from a predefined set. The drawback of these approaches is that they require rarely available labeled data and fixed lists of possible topics of interest.

As an alternative, some studies utilize unsupervised topic modelling methods, such as LDA [166], which represent inferred interests as a bag of related words. Such methods do not produce specific values or categories for the interests, making them inapplicable for building a personal knowledge base.

Several studies use external knowledge bases (e.g. Wikipedia) to link the users' content to the relevant concepts, which serve as the inferred interest value. Some approaches capture named entities in the social media submissions and link them to the corresponding Wikipedia pages [65, 105], which requires detecting explicit mentions. Alternatively, the whole text of a submission can be mapped to the Wikipedia category [142], determining the interest of the user. The disadvantage of these approaches is their reliance on an external knowledge base, which might be unavailable or require additional preprocessing, or may be short of required concepts.

2.1.2 Interpersonal relationships

There is only limited research on relationship prediction in dialogues, as most studies focus on literary texts. The relationships in novels are often predicted on the coarse granularity (positive or negative sentiment) [16], modelled as emotion-related classes (*anger*, *fear*) [67], or described in a topic-modelling manner [17, 58]. While fictional texts often contain dialogues, they are interleaved with narratives, where the language is less colloquial and more descriptive, which aids explicit extraction of fictional characters' relationships.

On the other hand, screenplays or scripts of theatre plays, movies or TV series are more similar to real-life conversations. Nalisnick and Baird [110] explored Shakespeare plays to analyze the polarity and intensity of emotions of characters towards each other. The same data is used in Azab et al. [3], where fine-grained relationship classes adopted from Massey et al. [101] are predicted by applying a logistic regression classifier on a pair of learned character embeddings. However, such approach predicts relationships solely based on characters' latent attributes without considering any conversational context.

Rashid and Blanco [132] investigated the prediction of 9 *interpersonal dimensions* (e.g., *intimate*, *intense* or *pleasure-oriented*) [169] of utterances in the Friends series. The authors trained bag-of-words SVM classifiers for each relationship dimension, to determine whether an utterance expresses, for instance, *equal* or *hierarchical* relationship. Similarly, Qamar et al. [125] leveraged vector representations of emotion words, to classify a dialogue taken from a movie script corpus into four attachment styles (e.g., *friend*, *family*) and four association types (e.g., *secure*, *fearful*), which are then combined into 16 relationship classes. Both approaches do not provide explicit and detailed information about the speakers' relationships, such as who is the *parent* of whom, and instead focus on relationship characteristics.

Speakers' relationships are part of 36 predicates investigated by Yu et al. [178], which focused on the general relation extraction task between two arguments appearing in a dialogue (e.g., *spouse*, *place_of_residence*), taken from the Friends series; 14 of the predicates refer to the relationships between people.

2.2 Methodology

This section describes technical details of the methods employed in this dissertation. We provide details on convolutional and attention-based neural models, describe zero-shot learning paradigm and document ranking task, define the metrics for model evaluation.

2.2.1 Neural networks

A neural network performs a series of transformations of the input data to get the desired output. The work unit in the neural network, a perceptron, computes a weighted sum of the input vector x:

$$z = f(Wx + b) \tag{2.1}$$

where W and b are the network parameters, which are updated during training. f is the *activation function*, adding non-linearity to the outputs. Popular choices for the activation

function are *sigmoid* and *Rectified Linear Unit* (ReLU) functions:

$$sigmoid(z) = \frac{1}{1 + e^{-z}} \tag{2.2}$$

$$ReLU(z) = max(0, z) \tag{2.3}$$

The neural network is comprised of multiple perceptrons, arranged in layers, performing sequential operations on the input. The outputs from the last layer of the network serve as the predictions of the model. For example, in multi-class classification the network's outputs can represent the scores for each predicted class. To transform the scores into probabilities of each class the *softmax* function is used:

$$softmax(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \tag{2.4}$$

During training the network learns how to reproduce the true values $\hat{y}$ by adjusting the trainable parameters W of each perceptron. To achieve that, the network's output y is compared with the desired $\hat{y}$ using a *loss function*, which evaluates how different are y and $\hat{y}$. In classification problems the most commonly used loss function is the *cross entropy* loss:

$$L = -\sum_{i=1}^{m} \hat{y}_i \log(y_i) \tag{2.5}$$

where m is the number of classes in the classification problem. The objective of neural network training is to minimize the loss function.

The process of adjusting the parameters of the network layer by layer is called *backpropagation*. The parameters are updated in the direction of loss minimisation by taking partial derivatives:

$$w' = w - \alpha \frac{\delta L}{\delta w} \tag{2.6}$$

where α is the *learning rate*, the speed of changing the weights. Learning rate is a hyperparameter of the network, which does not get updated in the backpropagation and needs to be carefully selected. Very high learning rates lead to parameters oscillating around the optimal values, while very small learning rates cause slow training convergence.

2.2.2 Word embeddings

Textual data used in natural language processing tasks can not be directly processed by neural networks, which require numerical data as input. To overcome this, usually each word in the vocabulary is encoded into a real-valued vector representation, called a *word embedding*. Word embeddings capture word's semantic information, so that the words which are synonymous or are used in similar contexts will have similar representations (e.g., in terms of cosine similarity of the corresponding embedding vectors).

Word embeddings can be learned jointly with neural network training. In this case each words is associated with a random real-valued vector, which is updated during the backpropagation in the network to capture task-specific work characteristics. Another popular option is to use pretrained word embeddings, learned by some external model.

One popular choice of pretrained word embeddings is *word2vec* [107] method, which preserves contextual information of the terms. Word2vec embeddings are trained on one of the tasks: *(i)* continuous bag of words (predicting the current term from its context) or *(ii)* skip-gram model (predicting surrounding terms given the current word).

The limitation of using pretrained word2vec embeddings is that the final embedding for a term is the same for all contexts this term can occur in, which can cause problems with the polysemous words (like '*bank*' or '*lie*'). Thus, recently the *contextualized* embedding models, such as BERT, described in Section 2.2.4.2, have recently gained considerable popularity. These models are pretrained on a large linguistic corpus, which enables them to statistically learn the structure of language. They produce the embeddings for the whole sequence simultaneously, effectively capturing terms' context.

2.2.3 Convolutional Neural Networks

A Convolutional Neural Network (CNN) [76] is a neural architecture, which is tailored for the tasks, where absolute positions in the input do not matter, i.e. the properties of subparts of the inputs to a CNN should be invariant to shifts. An example of such application is spam detection, where the task is to check if there are any suspicious phrases in the input, but the exact absolute position of such a phrase does not matter.

The input to a CNN is a word sequence $x_{1:n}$ of length n, where each word is represented by a k-dimensional embedding, $x_i \in \mathbb{R}^k$. The CNN multiplies a weight matrix $w \in \mathbb{R}^{hk}$, called a *convolutional filter*, to the slices of the input of length h. After applying the filter to each possible slice in the input, we get a *convolutional feature map*:

$$\mathbf{c} = [c_1, ..., c_{n-h+1}],$$

$$c_i = f(w \times x_{i:i+h-1} + b)$$

where $x_{i:i+h-1}$ denotes a slice between i-th and and $i+h-1$-th words, f is the activation function, c_i is a feature produced by a single filter application. Effectively the filter w detects if a h-sized slice has a particular feature (for instance, it can detect if a bigram $x_{i:i+1}$ is a pair "*verb + noun*"). The features in the feature map $\mathbf{c}$ are combined with some aggregation operation, for example, max-pooling $\hat{\mathbf{c}} = max(\mathbf{c})$, the result of which acts as a representative for the used filter (e.g. $\hat{\mathbf{c}}$ can show the maximum probability score of having spam in some input slice). On each network layer multiple filters are applied to detect various features, capturing different aspects of the input.

Having gained popularity in computer vision, CNNs have also shown strong results in natural language tasks, such as sentence classification or sentiment prediction [25, 62, 71]. Their advantages are a small number of parameters (the same small weight matrix is applied to each input slice), fast computation and good interpretability. However, the contexts considered by CNNs are limited by the selected convolutional filter size, therefore CNNs can not properly model long-term word dependencies.

2.2.4 Attention models

Attention mechanisms have been introduced to represent the long range contextual information more adequately, as compared to convolutional or recurrent architectures [6]. Attention assigns weights to the input elements, indicating which inputs to focus on to make a correct prediction. In natural language processing, attention mechanism is used to produce representations for sequence pairs (for example, in language generation) or single sentences (e.g., for sentence classification tasks).

Given as input a vector s and a context $h_1, ..., h_n$, the attention model creates a refined representation of s by incorporating the information from the context, based on the relevance of each context element h_i to the given s. For each h_i self-attention computes an *attention weight* α_{ij} as follows:

$$\alpha_i = \frac{exp(e_i)}{\sum_{k=1}^{n} exp(e_k)}$$

$$e_i = f(s, h_i)$$

where n is the length of context sequence and f is an attention function. f can be implemented with a dot-product, sum or be computed with a neural network.

Then the attention context vector for s will be a sum of the context elements h_i weighted by their attention scores:

$$c = \sum_{i=1}^{n} \alpha_i h_i \tag{2.7}$$

The context vector c can be used to augment the representation of s, for example, by concatenating them. Attention models were shown to work well for many natural language tasks, where distant contextual information is important [6, 177]. Moreover, attention computation can be effectively parallelized, as opposed to computations in sequential models. Attention weights are also highly interpretable, allowing to investigate which sequence elements were influential for creating contextual representations.

2.2.4.1 Transformer encoder

Transformer [158] is a state-of-the-art neural model based on attention mechanisms. Transformer is a sequence-to-sequence model, used to convert the input sequence to the output sequence (for example, it can be used for machine translation). The use of attention mechanisms allows Transformer to produce bidirectional input representations - capturing both left and right context of the tokens, as opposed to unidirectional representations generated by recurrent models.

Transformer consists of an *encoder*, for creating the representation of the input sequence, and a *decoder*, for generating the output. The encoder in Transformer can also be used as a standalone model for creating refined input sequence embeddings. In the following we will provide a description of the architecture of a Transformer encoder.

One Transformer encoder module consists of two blocks: a multi-head self-attention and a fully-connected feed forward neural network. Each block is surrounded by a residual

connection [55] (which adds the unchanged input to the output of the block), which is followed by layer normalization [4]. Transformer encoder is composed of several such modules stacked on top of each other.

The input elements to the encoder are summed with sinusoidal positional encoding, since attention mechanism does not preserve the information about the absolute positions. Self-attention function in Transformer, called *Scaled Dot-Product Attention*, estimates the compatibility of each word in the sequence (denoted as query Q) to the rest of the words (keys K), which forms coefficients in the weighted sum of the values (V):

$$attention(Q, V, K) = V \cdot softmax(\frac{QK^T}{\sqrt{d_k}}) \tag{2.8}$$

where d_k is the dimentionality of keys K, which scales the dot-product. In Transformer encoder, Q, K, V are all the representations of the input elements.

To capture the information from different subspaces from the input, in Transformer several attention functions are performed in parallel on different input projections, which is called *muli-head attention*. The Q, K, V matrices are linearly projected h times, where h is the number of attention heads, and attention function is applied to each projection:

$$head_i = attention(QW_i^Q, KW_i^K, VW_i^V). \tag{2.9}$$

where W_i are projection matrices. All heads are then concatenated and projected again:

$$MultiHead(Q, K, V) = concat(head_1, ..., head_h)W^O. \tag{2.10}$$

where matrix W^O maps from the projected dimension $h \cdot d_k$ back to the original dimension d_k.

The second block of the encoder is a two layer position-wise feed forward network with a ReLU activation between the layers:

$$ffn(x) = ReLU(xW_1 + b_1)W_2 + b_2 \tag{2.11}$$

The output of an encoder module serves as the input to the next one, which performs same operations on the input. After the last encoder module the output representation can be used for further tasks; e.g., it can be passed to a Transformer decoder or a classification layer.

2.2.4.2 Bidirectional Encoder Representations from Transformers (BERT)

BERT [32] is a model for creating deep contextualized representations of the input text. BERT is usually pretrained on a very large unlabeled text corpus and then can be fine-tuned for inference on other natural language tasks.

BERT is comprised of an embedding layer, several stacked Transformer encoder layers and a task-specific output layer. BERT is pretrained on two tasks: *(i)* masked language modelling and *(ii)* next sentence prediction.

In the *masked language modelling* task some tokens in the input are replaced with a special [MASK] token. Given the information from the context of the masked input, BERT

predicts the original value of the word, by producing the probability distribution over vocabulary. In the *next sentence prediction* task BERT is trained on pairs of sentences, trying to classify whether the second sentence was subsequent to the first one in the original document. Both tasks do not need explicit labels in the corpus, making pretraining of BERT semi-supervised.

BERT was shown to achieve state-of-the-art results on many NLP tasks [32], still it has several limitations. The number of trainable parameters in BERT is very large (110 million parameters in the BERT_{base} model and 345 million in BERT_{large}), which makes training it on a large corpus very time-consuming. Moreover BERT's input cannot exceed 512 tokens, which is too few for many tasks dealing with long texts. Trying to overcome this limitation leads to splitting or cropping the input, which causes loss of information.

2.2.5 Zero-shot Learning

The term zero-shot learning refers to the models making predictions over the set of labels, which may have been unobserved during training time. Zero-shot learning is applied to the problems with scarce training data, where the labeled training instances cover only few of the possible classes and test data can potentially contain unseen classes. One example of such problem is classification in a dynamic environment, where new classes appear within time; another example is a very fine-grained classification into a huge number of classes, where capturing sufficient training samples for each class is infeasible.

Usually zero-shot problems are solved by mapping training and test instances to some common latent representation space, where classification can be done easier. This mapping is learnt from the observed classes at training and is applied to the zero-shot classes at test time.

2.2.6 Document ranking

Information Retrieval (IR) addresses the task of searching through a document collection and retrieving the documents, which are relevant to the given query. It involves creating representations of both the documents (known as *indexing*) and the user's query, and defining a matching function between these representations, called a *ranking model*.

Indexing involves such operations as stemming, removing stop-words or creating inverted index (the mapping from each vocabulary word to the documents it occurs in together with the frequency of occurrence).

The ranking model outputs the optimal ranking of the documents with respect to the given query. Ranking models range from boolean, vector or probabilistic methods to neural models. In this section we give an overview of two ranking models we used in our research.

BM25 [135] is a popular probabilistic ranking model, based on the frequencies of query terms appearing in each document. For a query $Q = q_1, ..., q_n$ and a document D, BM25 produces a score:

$$score(D, Q) = \sum_{i=1}^{n} IDF(q_i) \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1(1 - b + b\frac{|D|}{avg\_doc\_len})} \tag{2.12}$$

where $f(q_i, D)$ is the frequency of occurrence of the term q_i in the document D, $|D|$ is the length of the document D and avg_doc_len is the average length of the document in the collection. The parameters k_1 and b can chosen freely. k_1 parameter regulates how much the document score can be affected by a single term; b controls the impact of the relative document length. $IDF(q_i)$ is the inverse document frequency of the term q_i, which can be calculated as:

$$IDF(q_i) = ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right) \tag{2.13}$$

where N is the number of documents in the collection, $n(q_i)$ is the number of documents containing q_i. Inverse document frequency effectively assigns lower scores to the terms that are frequent in the document collection.

BM25 formula is intuitive and shows good results, even compared to the state-of-the-art neural approaches. However, it requires exact term matches and does not consider the order of the words in the query and the document.

K-NRM [173] is a neural ranking model with takes advantage of embedding similarity between the query and document representations, addressing the limitations of the models with exact term matches. K-NRM creates a translation matrix, containing the similarity between each pair of query and document words. After that a kernel-pooling technique is applied to extract soft match features (soft count of word pairs frequencies at multiple similarity levels), which are used to produce the final ranking score. The kernel method helps to handle the imprecise word matches, caused by calculating word similarity at a single level.

2.2.7 Evaluation metrics

In this section we discuss classification and ranking quality metrics used for model evaluation in our experiments. Depending on the nature of the predicted attribute (binary or multi-class; single- or multi-label) we selected corresponding metrics.

Accuracy, recall, precision, F1. We first give the definitions for the classification metrics in the basic binary case (positive or negative class):

$$accuracy = \frac{\text{num correct predictions}}{\text{num test instances}} \tag{2.14}$$

$$precision = \frac{TP}{TP + FP} \qquad recall = \frac{TP}{TP + TN} \tag{2.15}$$

where TP, *true positives*, is the number of true labels that the model correctly predicted; TN, *true negatives*, is the number of true labels that the model failed to predict; FP, *false positives* is the number of incorrect models' predictions. Precision shows how accurate model's predictions are, recall shows how many true labels the model could identify.

The output of the binary classifier depends on the score of the positive class: if the score exceeds a particular *decision threshold*, the positive class is predicted, otherwise, the negative. If the model scores classes with probabilities, the default threshold is usually set

at 0.5; yet, depending on the model architecture and loss function used for training, it may be beneficial to tune the exact value of the threshold. Varying decision threshold results in changing TP and FN error counts (changing the number of instances to be classified as positive), which causes one of precision or recall metrics to increase and the other one to decrease. It is therefore useful to calculate their harmonic mean:

$$F1 = \frac{2 \cdot precision \cdot recall}{(precision + recall)} \tag{2.16}$$

AUROC. A *Receiver Operating Characteristic* (ROC) curve is a plot depicting the performance of the binary classifier at varying decision threshold values. ROC curve is plotted in true positive rate (TPR) vs. false positive rate (FPR) coordinates. TPR is equal to recall; FPR is calculated as:

$$FPR = \frac{FP}{FP + TN} \tag{2.17}$$

AUROC metrics computes the area under the ROC curve. It can be interpreted as an expectation that a randomly drawn positive example will be ranked higher than a randomly drawn negative example. AUROC is a binary classification metrics and its extension to the multi-class case requires binarizing the predictions.

Multi-class metrics. For multi-class predictions, precision, recall and F1 metrics can be computed using *micro* or *macro* averaging. In case of micro averaging the metrics are calculated globally across all classes; macro averaging implies calculating metrics per class and averaging them. For example, the equations for micro and macro precision for classification with N classes will look like:

$$precision_{micro} = \frac{\sum_{c=1}^{N} TP_c}{\sum_{c=1}^{N} TP_c + \sum_{c}^{N} FP_c} \tag{2.18}$$

$$precision_{macro} = \frac{1}{N} \sum_{c=1}^{N} \frac{TP_c}{TP_c + FP_c} \tag{2.19}$$

A *confusion matrix* is an $N \times N$ matrix which shows the number of misclassifications between each pair of classes. The rows correspond to the true class labels and the columns to the predicted labels; the number in the ij-th matrix cell denotes the number of times the model predicted class j, when the correct label was class i. Confusion matrix is useful for qualitative model analysis, allowing to notice systematic misclassifications.

2.2.7.1 Ranking metrics

Multi-class classification can also be viewed as a ranking problem, where the aim is to assign the highest rank to the correct label in the list of class scores outputted by the model.

nDCG. A normalized Discounted Cumulative Gain (nDCG) measures the gain of placing each label on its position in the ranked list of model's predictions, based on labels' relevance. For classification problems we can set the relevance of all correct labels to 1, and 0 for incorrect ones. A Cumulative Gain at rank r (CG_r) is calculated as:

$$CG_r = \sum_{i=1}^{r} rel_i \tag{2.20}$$

where rel_i is the relevance of the class at the position i in the ranked list of predictions. CG only calculates the total relevance of predictions up to position r, irrespective of their order. The Discounted Cumulative Gain (DCG) reduces the relevance impact of the correct predictions logarithmically proportional to their position in the list:

$$DCG_r = \sum_{i=1}^{r} \frac{rel_i}{log_2(i+1)} \tag{2.21}$$

A normalized Discounted Cumulative Gain (nDCG) is a fraction of the achieved DCG to the ideal one:

$$nDCG_r = \frac{DCG_r}{IDCG_r} \qquad IDCG_r = \sum_{i=1}^{|rel_r|} \frac{rel_i}{log_2(i+1)} \tag{2.22}$$

where $IDCG_r$ is an ideal DCG, calculated on the ranked list of only relevant results up to the position r. For a perfect ranking model $nDCG_r = 1$.

MRR. The mean reciprocal rank (MRR) is the ranking metrics, calculating the average of the inverse ranks of correct predictions in the sorted list of results:

$$MRR = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{rank_i} \tag{2.23}$$

where N is the number of test instances. MRR considers only the rank of the first correct label; thus, it is not applicable to the multi-label classification.

2.2.8 Significance tests

The best performing model is usually selected based on comparing the evaluation metrics computed on the models' predictions on the same test set. However, the difference in performance between two models can be caused by statistical fluctuations; *significance tests* are used to eliminate the possibility of this.

The null hypothesis to test is that the mean difference between paired statistics of the two given models is zero. Here we compare the arrays of predictions from the considered two models, which are aligned so that each pair of predictions is calculated on the same test sample.

2.2.8.1 Paired t-test

Paired t-test is the most widely used significance test in machine learning. To compute sample-based t-test for a specific metrics x, we first need to compute this metrics on each individual sample i. For example, in case of MRR metrics, one prediction of the model A will be:

$$x_i^A = \frac{1}{rank_i} \tag{2.24}$$

Let $x_i = (x_i^A - x_i^B)$ be the difference of statistics on predictions of models A and B for an instance i. We compute the average and standard deviation of the differences between all pairs:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})}{n-1}} \tag{2.25}$$

where n is the number of test samples. Then t-statistics for the Student's t-test can be computed as follows:

$$t = \frac{\bar{x}}{\sigma\sqrt{n}} \tag{2.26}$$

After that, we find the *p-value* - the probability of obtaining test statistic at least as extreme as the observed statistic under the null hypothesis: $p = Pr(T \geq t|H_0)$. Under the null hypothesis, the obtained statistic follows a *t-distribution* with $(n-1)$ degrees of freedom; the probability $Pr(T \geq t|H_0)$ can be looked up in statistical tables for t-distribution.

Finally, the obtained p-value is compared to a pre-selected cutoff value (usually set to be 0.05). If computed p is less than the cutoff, the null hypothesis can be rejected and we can conclude that the results of models A and B are significantly different with respect to selected metrics x.

Sample-based t-test can be computed for micro averaged evaluation metrics; in the case of macro averaging, the paired arrays of evaluation metrics for the two models represent the metrics computed on each class, as opposed to each test sample. For instance, for macro MRR the computation for each class i will be:

$$x^A_{class_i} = \frac{1}{n_i}\sum_{j=1}^{n_i}\frac{1}{rank_j} \tag{2.27}$$

where n_i is the size of class i and each instance j in the summation has class i as the true label. After that all calculations of t-statistic (Equations 2.25-2.26) and p-value are the same as in the sample-based case.

2.2.8.2 McNemar's test

When the number of samples (classes) serving as input to t-test is small (which is topical for macro-averaged metrics on a small number of classes), t-test might yield imprecise results (as $\sqrt{n}$ directly influences the computation of t-statistic). Instead, for such cases one can use *McNemar's test* [33] to evaluate the significance of the difference in two models' predictions.

McNemar's test is based on 2×2 *contingency table* of the counts of misclassifications for models A and B, computed on matched pairs of samples.

	model A correct	model A incorrect
model B correct	n_{00}	n_{01}
model B incorrect	n_{10}	n_{11}

Table 2.1: Contingency table for McNemar's test.

For example, the value n_{11} from Table 2.1 denotes the number of samples both models A and B classified incorrectly. The null hypothesis in McNemar's test is that the number of misclassifications for both models is the same, i.e., $n_{01} = n_{10}$. The test uses χ^2 statistics:

$$\chi^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{(n_{01} + n_{10})} \tag{2.28}$$

Under the null hypothesis the computed statistics follows χ^2 distribution with 1 degree of freedom; the p-value can be found from corresponding tables for χ^2. Similarly to the t-test case, the significance of experiment is proven by comparing the obtained p-value to the selected cutoff value.

Chapter 3

Conversational Datasets

Contents

Training supervised models for personal attribute prediction requires significant amounts of labeled data. Although there is plenty of available dialogues for training chat-bots, the speakers' profiles containing personal information are rarely accessible. There is only a limited number of publicly available conversational datasets labeled with speakers' attributes, mostly containing only mere basic demographic facts, like *gender* or *age*. To efficiently train models for inferring rich user profiles we create large scale datasets containing a wide range of personal attributes.

Our work is focused on personal attribute prediction from the *textual* representation of the dialogues. We distinguish different sources of conversational data in textual format: spoken dialogue transcripts, private messages and emails, posts in web discussion forums, etc. In this chapter we describe our work on collecting and labelling *(i)* transcribed dialogues, and *(ii)* social media submissions. All discussed datasets are available at http://pkb.mpi-inf.mpg.de.

3.1 Background

In this section we elaborate on the methods for manual data labeling and evaluation of the obtained results. To annotate the datasets described in Section 3.2 we turn to crowdsourcing using MTurk online platform.

Crowdsourcing is a powerful tool for getting annotations for large volumes of data at a low cost. Crowdsourcing is the process of solving a task by collecting and aggregating opinions from a group of people, called *annotators*. Each annotator does not have to be an expert in the given task; a reasonable quality of the results is achieved by aggregating the annotations of a large number of people, e.g. using majority voting. The details on various answer aggregation approaches will be given in Section 3.2.3.3.

Mturk crowdsourcing platform. Amazon Mechanical Turk (MTurk) is a crowdsourcing online tool, which enables researchers to publish surveys for data collection. The published Human Intelligence Tasks (HITs) are usually simple, so that they can be completed within a short time by a human annotator, yet HITs are hard for automated methods (for example, psychological surveys). Each completed HIT comes with metadata, such as the annotator's id or completion time, allowing to maintain the annotation quality by filtering out regularly underperforming workers. MTurk has proven to be a reliable tool, providing cheap data annotations [115] used by researchers in many scientific areas. Crowdsourcing on MTurk allows the access to a more diverse demographics of annotators, as opposed to hiring live participants.

Inter-annotator agreement metrics. Evaluating the reliability of crowdsourced annotations using inter-annotator agreement metrics is an important step to ensure high quality of the collected labels. Additionally, calculating the degree of agreement allows to get an insight into the difficulty of the annotation task, which is helpful for modelling and refining the crowdsourcing assignments.

Fleiss' kappa [39] is a statistical inter-rater agreement measure applicable to the annotations for the multi-class classification problems. This metric allows any fixed number of annotators per item and evaluation of each item can be done by a different set of annotators. Fleiss' kappa computes the degree of the obtained inter-rater agreement over the agreement expected by chance.

Let N be the number of annotated pairs, indexed by $i = 1, ..., N$; K be the number of classification labels, indexed by $j = 1, ..., K$; and n_{ij} be the number of annotators, who assigned j-th label to the i-th item. We first calculate the agreement of annotators per item P_i and the proportion of items per label p_j:

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^{K} n_{ij}(n_{ij} - 1)$$

$$p_j = \frac{1}{\sum_{i=1}^{K} k_i} \sum_{i=1}^{N} n_{ij}$$

Then Fleiss' kappa κ is calculated as follows:

$$\kappa = \frac{\tilde{P} - \tilde{P}_e}{1 - \tilde{P}_e} \tag{3.1}$$

where

$$\tilde{P} = \frac{1}{N} \sum_{i=1}^{N} P_i \qquad\qquad \tilde{P}_e = \sum_{j=1}^{k} p_j^2 \tag{3.2}$$

The denominator in the equation for κ shows the degree of agreement reachable above chance; the numerator shows the agreement that has actually been achieved. $\kappa = 1$ denotes perfect inter-rater agreement.

To account for the multi-label case we modified Fleiss' kappa calculations as follows: for each item $i = 1, ..., N$ we denote k_i to be the number of labels which were selected by at least one annotator for the item i. To calculate multi-label Fleiss' kappa we only have to change the equations for P_i and p_j:

$$P_i^{multi} = \frac{1}{k_i n(n-1)} \sum_{j=1}^{K} n_{ij}(n_{ij} - 1) \tag{3.3}$$

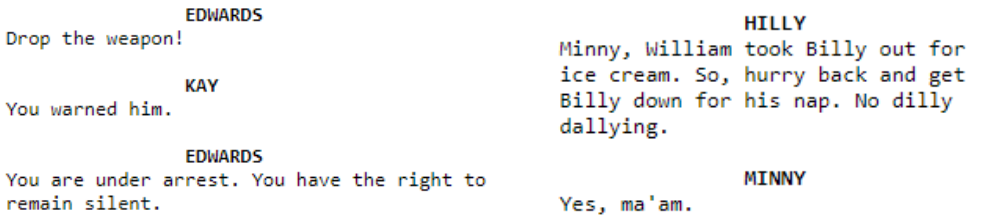
$$p_j^{multi} = \frac{1}{\sum_{i=1}^{K} k_i} \sum_{i=1}^{N} n_{ij} \tag{3.4}$$

leaving the equations 3.1-3.2 the same as before. The single-label Fleiss kappa variant is used in Section 3.2.2.2 to evaluate the manual annotation of the *profession* attribute and in Section 3.3.3.6 to validate the precision of the weakly-supervised labeling approach; we use multi-label kappa in Section 3.2.3 to estimate the agreement on crowdsourced labeling of inter-speaker relationships.

3.2 Movie script dataset

Following prior work on personalized dialogue systems, we explore the applicability of fictional dialogues from TV or movie scripts to approximate real-life conversations [83, 89]. Movie scripts are a practical source of transcribed conversations because they are freely available, dialogue-intensive, and each utterance has the speaker marker (unlike, for example, dialogues in novels). Exemplary conversations from popular movies are shown in Figure 3.1.

Specifically, we created two datasets consisting of characters' utterances: Movie Character Attributes dataset (MovieChAtt), labeled with speakers' demographic attributes (*age*, *gender*, *profession*), and Film Relationship dataset (FiRe), labeled with characters' *interpersonal relationships*.



Excerpt from "Men in Black" (1997) **Excerpt from "The Help" (2011)**

Figure 3.1: Examples of conversations in the movie scripts.

3.2.1 Related work

In this section we give a brief overview of the existing conversational datasets of speakers' personal attributes and interpersonal relationships. We outline their shortcomings, which we addressed by collecting our own datasets.

3.2.1.1 Datasets of demographic traits

Many conversational datasets labeled with personal attributes are not publicly available due to privacy protection of the speakers' data. The few accessible datasets include spoken dialogue transcriptions [23, 93] and artificially created conversations [61, 183].

Cieri et al. [23] gathered transcribed telephone dialogues on the given general topics; each speaker indicated their age, gender, dialect, education and occupation. Love et al. [93] created a corpus of transcribed casual conversations of British English speakers, where all speakers specified their demographic (*age*, *nationality*, etc) and linguistic attributes (*mother tongue*, *dialect*).

Joshi et al. [61] introduced an extension to the bAbI goal-oriented user-chatbot dialogue dataset [12], providing the users' ages, genders and food preferences. Zhang et al. [183] created a crowdsourced conversation dataset, Persona-Chat, where each speaker had to employ the given personality, described with a few sentences. A drawback of Persona-Chat is that it provides only textual descriptions of personas, as opposed to precise demographic facts. In general, the conversations created in a controlled way sound artificial, lacking of natural topic drifts and unnecessarily emphasizing the required content.

3.2.1.2 Datasets for relationship prediction

Two popular sources of conversational data for interpersonal relationship inference are literary texts and movie scripts. Several studies provided annotated relationships of the characters in novels as binary labels [16] (positive or negative sentiment) or described as bags-of-words [58]. Massey et al. [101] annotated the characters in literary texts with relationships on different granularity, additionally indicating the temporal change in relationship.

Compared to literary texts, movie and series scripts provide dialogues in a structured format, simplifying speakers' identification. Chen et al. [20] collected conversations from Chinese TV series scripts and used three annotators to label them with 24 relationships and 7 emotions. The relationship labels were hierarchically split by field (*family*, *school*, *company*, *other*) and seniority (*elder*, *peer*, *junior*). TV series scripts were also used by Yu et al. [178], where the script of Friends series was annotated by two judges with 36 predicates for relation extraction task, where 14 of the predicates indicated the relationship between people. Jia et al. [59] annotated relationships of the characters in the movie scripts with 13 relationship labels, belonging to four main categories (*family*, *intimacy*, *official*, *others*), resulting in the DDRel dataset.

Unlike most prior works, we consider *directed* relationships (e.g., *parent* and *child* as separate labels) and allow each speaker pair to have *multiple* relationship labels. Moreover, our crowdsourced annotation is based on the fine-tuned agreement among at least 6 annotators, which provides more reliable aggregated results than in most related works.

3.2.2 MovieChAtt dataset

To overcome the limitations of the existing datasets of transcribed conversations we introduce *Movie Character Attributes* dataset (MovieChAtt). MovieChAtt is based on a subset of characters in the Cornell Movie-Dialogs Corpus [28] consisting of 617 movie scripts. From each movie we derive a sequence of utterances for each character, excluding the characters who have less than 20 lines in the movie. We label the acquired characters with *age*, *gender*, and *profession* attributes. The details on the attribute value lists and label distributions are given in Table 3.1; the overall dataset statistics are given in Table 3.4. In the following we describe the labeling process for each personal attribute.

Age	Gender	Profession			
adult (2645)	female (959)	criminal (194)	writer (45)	manager (19)	airplane pilot (12)
middle-aged (1183)	male (1003)	military personnel (143)	unemployed (39)	banker (17)	nurse (12)
teenager (389)		student (84)	musician (36)	school teacher (17)	clerk (11)
senior (220)		child (83)	lawyer (32)	psychologist (17)	professor (11)
child (74)		special agent (77)	actor (32)	journalist (16)	photographer (9)
		businessperson (71)	politician (26)	waiter (16)	activist (9)
		policeman (66)	priest (24)	director (16)	engineer (8)
		housewife (66)	astronaut (24)	editor (15)	painter (7)
		doctor (63)	assistant (24)	salesperson (14)	explorer (5)
		scientist (58)	sportsman (20)	driver (14)	stewardess (4)
		detective (58)	monarch (20)	tv/radio presenter (13)	

Table 3.1: Lists of age, gender and profession attribute values in the MovieChAtt Dataset with value counts.

3.2.2.1 Labeling age and gender

We extracted characters' *gender* and *age* attributes by associating the characters with their entries in the Internet Movie Database (IMDb) and extracting the corresponding actor or actress' attributes at the time the movie was filmed, assuming that the gender of the actor and the character coincide in most cases.

This yielded 1,963 characters labeled with their genders and 4,548 characters labeled with their ages. We discretized the age attribute into the following ranges: (i) 0–13: *child*, (ii) 14–23: *teenager*, (iii) 24–45: *adult*, (iv) 46–65: *middle-aged* and (v) 66–100: *senior*. In our data the distribution of age categories is highly imbalanced, with *adult* characters dominating the dataset (58.7%) and *child* being the smallest category (1.7%).

3.2.2.2 Labeling professions

To obtain the ground-truth labels of characters' *profession* attributes, we conducted a Mechanical Turk crowdsourcing task to annotate 517 of the movies in our corpus. The workers were asked to indicate the professions of characters in a movie given the movie's Wikipedia article. The workers were instructed to select professions from a general predefined list if possible (e.g., *doctor*, *engineer*, *military personnel*), and to enter a new profession label when necessary. We manually defined and refined the list of professions based on several iterations of MTurk studies to ensure high coverage and to reduce ambiguity in the

Family	Social	Professional	
parent (41)*	friend (208)*	colleague/co-worker (67)*	boss/employer/master (29)*
child (48)*	enemy (27)*	doctor/patient (medical, 19)*	employee/servant (34)*
sibling (37)*	(ex-)love interest (lover, 187)*	client/seller (commercial, 19)*	religious relationship
(ex-)spouse (69)*	fan	classmate	
engaged	idol	teacher	
distant family member	members of the same club	student	

Table 3.2: List of relationship labels split into categories. Labels marked with * are included in the final dataset and are supplied with number of acquired pairs.

options(e.g., *journalist* vs *reporter*). We also included non-occupational "professions" that often occur in movies, such as *child* and *criminal*.

Fleiss' kappa for the crowdworkers' inter-annotator agreement is 0.47. Disagreement was oftentimes caused by one character having multiple professions (Batman is both a *superhero* and a *businessman*), or a change of professions in the storyline (from *banker* to *unemployed*). We kept only characters for which at least 2 out of 3 workers agreed on their profession, which yielded 1405 characters labeled with 43 distinct professions. The highly imbalanced distribution of professions, shown in Table 3.1, reflects the bias in our movie dataset, which features more *criminals* and *detectives* than *waiters* or *engineers*.

3.2.3 FiRe dataset

Addressing the need for a relationship dataset with *directed, multi-label* interpersonal relationships of the conversation interlocutors we issue *Film Relationship* dataset (FiRe). Compared to similar datasets, FiRe provides fine-grained relationship annotations, allowing multiple directed relationship labels per speaker pair.

3.2.3.1 Data preparation

We use the *Jinni Movie Dataset* collected in Gorinski and Lapata [47], which provides speaker labels for each utterance as well as the film genre metadata. We selected the movies which:

- can be automatically associated with their Wikipedia page for annotation purposes
- have real-life genres, such as *drama* or *family*, to better approximate real-life conversations.

The selection of realistic movie scripts distinguishes FiRe from other character relationship datasets, such as in Jia et al. [59]. The model trained on FiRe is potentially more adaptive to real-life dialogues.

For each pair of characters we kept only the film scenes where they are the only participants. Additionally, we include all uninterrupted dialogue spans of the considered pair in the scenes with exactly three characters. We kept only the pairs which have at least 30 utterances throughout the whole movie.

	partial accuracy	total accuracy	precision	recall
MV	**0.98**	**0.68**	**0.88**	0.76
GLAD	**0.98**	0.67	**0.88**	0.76
DS	0.97	0.59	0.79	0.82
BCC	**0.98**	0.67	0.83	**0.85**

Table 3.3: Comparison of answer aggregation methods.

3.2.3.2 Crowdsourcing annotation

Inspired by Massey et al. [101], we manually created a list of 21 fine-grained relationships, divided into three categories: *Family*, *Social* and *Professional* (Table 3.2). We annotated character pairs in our dataset using MTurk, following the task design described in Massey et al. [101]. For each character pair a worker was supposed to indicate all applicable relationships, given the links to the movie descriptions (Wikipedia and GradeSaver, if available). Based on several pilot runs we opted to assign the labels agreed by 4 out of 6 annotators.

3.2.3.3 Label aggregation

We selected the best label aggregation method based on the evaluation of several state-of-the-art models, ranging from the basic majority voting to the more complex resource-intensive methods. To create the ground truth for comparison, we manually annotated 15% of the pairs, retaining the labels on which 2 out of 3 annotators agreed.

We used a Crowdsourcing benchmark by Zheng et al. [187], implementing state-of-the-art aggregation approaches, which enabled us to try different aggregation methods. We report the results of the best performing ones:

- *David Skene model* (DS) [31] is based on Expectation Maximizaion algorithm (EM), which jointly estimates the expertise of workers and the task label. This method has shown consistently optimal performance in many studies.
- *Generative model of Labels, Abilities, and Difficulties* (GLAD) [167] is an extension to EM that additionally estimates the difficulty of each task.
- *Bayesian Classifier Combination* (BCC) [68] uses Gibbs sampling to optimize the posterior joint probability of labels and workers.

We compare them to the majority voting (MV) approach. Note, that most of the models are based on the assumption of single-label answers, so we had to reformulate the problem as multiple binary decision problems to fit them.

Taking into account that each pair can have multiple labels associated with it and that the agreement can be reached only on a subset of those labels, we propose to evaluate both *partial* accuracy (the workers' answers partially match the golden set) and *total* accuracy (the workers' answers and the golden set are identical). Additionally, we evaluate precision and recall for all approaches. The results are shown in Table 3.3.

The compared models show almost equal performance, with MV having the greatest total accuracy and BCC yielding the best recall. We opted to use MV aggregation, as we

	MovieChAtt		FiRe		Series	
	avg	**max**	**avg**	**max**	**avg**	**max**
words per utterance	11	556	13	602	13	340
utterances per pair	79	471	99	597	417	15,216
words per pair	823	7798	1,087	3,977	6,562	188,676

Table 3.4: Statistics for MovieChAtt, FiRe and Series datasets.

consider high precision and accuracy more important for this task; additionally, MV is easier to interpret. One reason why the iterative approaches do not outperform simple majority voting is that most MTurk workers label only 1-2 samples, which is too few for the iterative models to effectively infer the workers' expertise.

To further ensure the high quality of our annotated data, we used the *Honeypot method* [77], where the questions with the known true answers (honeypots) are mixed into the task. The workers' scores are calculated as the fraction of their correct answers to the honeypots; the workers who did not get any honeypots were assigned an average score. After that all workers' answers are scaled by the obtained scores and the label was considered as correct if the sum of its votes exceeded a threshold, finetuned on the annotated set.

3.2.3.4 Dataset analysis

We obtained a multi-label Fleiss kappa of 0.45, which corresponds to moderate agreement. In total we collected 783 annotated character pairs from 254 films, of which 5% are labeled with multiple relationships. The original set of labels was filtered to include only those which have at least 20 representative samples, resulting in 12 labels. Summary statistics of the final dataset are given in Table 3.4 and the relationship label distribution in Table 3.2. We observed that the label distribution is heavily biased towards *friend* and *lover* labels, encountered almost three times more often than the third most popular label *spouse*.

3.2.3.5 Series dataset

We created an additional dataset of labeled TV series scripts, which are different from film screenplays, because they contain a longer history of interactions. The scripts of the series were crawled from IMSDb. As there is no information about scene boundaries in the gathered scripts, for each given character pair we kept only the uninterrupted sequences of at least 7 utterance turns.

For the resulting dataset we selected the series which would be realistic and diverse in topics. Following the same crowdsourcing annotation procedure as for FiRe, we collected 365 labeled pairs with 0.33 Fleiss' kappa agreement; the dataset statistics are included in Table 3.4. Compared to FiRe, character pairs in this dataset have larger number of utterances, around four times as much on average.

3.2.4 Discussion

Although the dialogues in films resemble real-life conversations, they sometimes sound artificial and allegorical, being produced from an existing script and well-rehearsed. Compared to real conversations, the interactions in the movies usually contain less colloquial speech, abbreviations and dialect words, so that they are more understandable to the general audience. Another drawback of using movie data is that many films have unrealistic elements in their plot, which can not be completely handled by our proposed genre filtering. Finally, the distribution of labels for some personal attributes in the movies do not follow those in real life (for example, big bias towards *lover/friend* relationships or heroic professions).

3.3 Social media submissions

Reddit is a popular social media platform for discussing a wide range of topics. It has become an prominent source of information for data analysis on social media as it provides an abundance of data with rich structure. Such data has many applications, including personalizing healthcare [51], recommendations, search, and conversational agents. Reddit is used by approximately 330 million users[1] with 2.8 million comments written each day[2].

Despite its popularity and abundance of data, few have considered Reddit as a source for inferring users' personal traits. However, many Reddit submissions contain a sufficient amount of personal information; an exemplary submission, indicating the user's hobby, is shown in Figure 5.2. Prior work has focused on Reddit merely as a source of demographic information, whereas rich attributes, like *profession* and *hobby*, are usually overlooked.

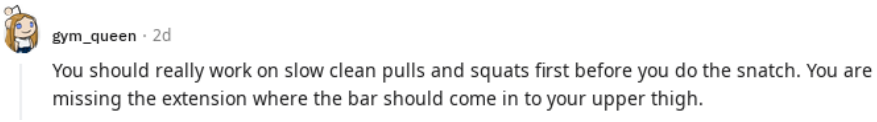


Figure 3.2: Example of a Reddit comment.

We address this gap by creating a labeled dataset of Reddit users (including their posts and comments) that covers five user attributes: *profession, hobby, family status, age,* and *gender* [153]. The collected submissions can be used as a proxy for dialogue utterances in conversational data research.

3.3.1 Related work

Automatic methods for identifying users' personal attributes from social media focus on user-generated content from Twitter, with a few exceptions that explore Facebook [138, 140] or Reddit [34, 38, 46] posts. Such methods, particularly supervised learning approaches, require a collection of user-generated content labelled with personal attributes of interest.

Data collection for such models is mostly done via: manual annotation after a focused search with specific keywords or hashtags [123, 130], public profile linked to Twitter profile

[1] https://redditblog.com/2018/11/13/holiday-on-reddit
[2] https://www.digitaltrends.com/social-media/reddit-ads-promoted-posts

description [14, 40], self-reports as part of an online survey [38, 40, 122, 124, 138, 141], or pattern-based extraction approach (e.g., `(I|i) (am|'m|was) born in + number (1920-2013)`) on user profile description or user posts [34, 70, 146, 152]. Several works [9, 10] made use of labelled datasets published within the shared task on author profiling organized by the CLEF PAN lab [41, 120].

There has been less effort on identifying demographic attributes of Reddit users compared with the body of work that exists for Twitter users. However Reddit posts have been exploited for other purposes, such as determining *users' personality* [46], *mental health condition* [24], *domestic abuse* [139] and *irony detection* [160], among others. Thelwall and Stuart [151] investigate how the topic of a subreddit influences the gender ratio within it. The study was performed on 100 subreddits grouped by interest; gender information about the users was collected by guessing it from their usernames, which is arguably a low-precision strategy. Smaller scale Reddit datasets exist for *gender*, *age* and *location* attributes [34, 38], which are unfortunately not publicly available. As far as we know, we are the first to consider *hobby* as a personal attribute of interest to be identified from online communication.

3.3.2 Background

Posts and Comments. Discussions on Reddit are organized in threads, which are initiated by an original *post* and may contain *comments* replying to the post and to the other comments. This creates a hierarchical structure that resembles a conversation between the users. Both posts and comments can be a textual content, a link with anchor text or images.

Subreddits. Reddit is organized into subreddits, which are fora that focus on specific topics. Those can be split by interest (sports, politics, etc), by country or community, type of content (text, gifs, videos), and so on. Subreddits have their own rules, but any registered user can create them. By convention, subreddits are prefixed with `/r`. For example, users discuss hockey in the `/r/hockey` subreddit.

Flairs. Flair is a user or post metadata that is a unique feature of Reddit. Flair is a small image with a short text description that is attached to a post or a username. Flairs can be defined differently for specific purposes by each subreddit. For example, in `/r/travel` subreddit they may indicate the *country* of the user, *gender* in `/r/AskMen` and `/r/AskWomen` or users' *favorite teams* in `/r/hockey`. Flairs for posts can be useful to filter and search for a particular content.

3.3.3 RedDust dataset

In this section we describe our proposed *RedDust* dataset, containing a collection of Reddit users. Each user in the dataset is associated with posts and comments they produce (which we call *submissions* in the following) and users' inferred personal attributes. We considered five personal attributes including *gender*, *age*, *family status*, *profession* and *hobby*. The dataset is created from the openly published Reddit dump, which spans between 2006 and 2018.

There are several criteria on which users and submissions are included in RedDust, i.e.,

users who posted between 10 and 100 submissions, and submissions containing between 20 and 100 terms after filtering. We filtered out hyperlinks and user mentions (i.e., `@nickname`) from the original content.

Some subreddits are likely to contain many false positives, such as those concerned with video games or role playing. This leads to personal assertions talking about the users' projected persona in a particular context (e.g., *"I am a priest looking for a guild"*). To mitigate this source of false positives, we blacklisted subreddits about gaming, fantasy and virtual reality from the top 500 subreddits sorted by the number of unique users. Posts made to blacklisted subreddits were discarded. Similarly, we discarded posts that contain quotations in order to reduce the possibility of the user referring to a third person (*"... and he shouted 'Hands on the counter, I am a cop!' "*).

For attributes that usually have a unique value (i.e., *gender*, *age* and *family status*) we also exclude users who state multiple different values to avoid introducing false positives. Meanwhile, we allow each user to have multiple attribute values for *profession* and *hobby*. The age of a given user is calculated relative to his or her age when writing the most recent comment. In the following we discuss particular techniques used to extract values for each personal attribute.

3.3.3.1 Gender

Gender has been the most popular user attribute to predict in existing user profiling work, particularly on Reddit [34, 151, 157]. In RedDust we consider gender as a binary predicate (*female* or *male*) as has been done in prior work.

Instead of considering usernames as a means for gender classification, as was done by Thelwall and Stuart [151], we look for self-reported gender assertions, which provide labels of higher precision. Specifically, we identified users' gender using the following methods:

- **Natural language patterns**. Following Fabian et al. [34], we manually created a set of patterns that indicate a specific gender. They have the general form of `(I am|I'm) a? <gender indicator>`, meaning that matches should contain '*I am*' or '*I'm*', optionally followed by an article '*a*', then a word that indicates gender like '*man*' or '*mother*'. A comprehensive list of patterns we used is given in Table 3.5, and the indicative gender words are shown in Table 3.6. Although the gender of a given user can be expressed in a longer snippet like "*I am a great mother*", we do not allow extra words like '*great*' to appear before gender-indicating words. This reduces false positives from statements like "*I'm a far cry from my mother*".

- **Bracket patterns**. In certain situations, users often volunteer to indicate their demographic information in order to give their posts more context ("*I [30f] was dating this guy [35m]...*"). This is common in relationship-related subreddits, where the users' age and gender are often relevant to the discussions. These cues are generally written in round or square brackets. To reduce false positives, we do not consider such patterns when they appear without brackets. To capture gender and age expressed in this way, we look for patterns of the form `(I|I'm|me) [<number>(m|f)]`.

attribute	pattern(s)
gender	`(I am|I'm) a? <gender indicator>` (e.g., *man*, *mother*)
age	(i) `I (was|am) born in <four digit year>` (ii) `I (was|am) born in <two digit year>` (iii) `I was born on <day, month, year>` (iv) `I am <number> years old` (v) `I am <number>` immediately followed by punctuation or conjunction
family status	(i) `I am <self-status indicator>` (e.g., *divorced*, *single*) (ii) `(my|I have a) <partner indicator>` (e.g., *wife*, *boyfriend*)
profession	`(I am|I'm) a <profession name>`
hobby	`<phrase indicator>` (e.g., *I enjoy*, *I like*) `<hobby name>`

Table 3.5: Patterns for labeling Reddit users with personal attributes.

- **Flairs**. Like Vasilev [157], we also consider gender-indicating flairs attached to users. This logic is subreddit-specific, so we restrict ourselves to common subreddits. For example, in subreddits `/r/AskWomen` and `/r/AskMen` the flair is one of *male, female, trans*, and so on, whereas in `/r/tall` and `/r/short` the flair is either *pink, blue*, or *other*.

3.3.3.2 Age

We label users' posts with age predicate using similar techniques as for gender:

- **Natural language patterns**. To infer users' age, we utilized five patterns listed in Table 3.5, with pattern (v) specifically designed to avoid false positives as in "*I am 6 feet tall*". We then calculated the exact age for patterns (i)-(iii) by subtracting the birth year from the publishing year of the post containing such patterns.
- **Bracket patterns**. Numbers indicating age were jointly collected along with gender, as described in the above-mentioned bracket patterns for gender.

Finally, we made sure that the obtained ages for users in RedDust are within the range of 10-100 years old, since users under 13 are not allowed to register and there are unlikely to be many users above 100 years old. This is helpful for reducing false positives, such as those in conditional sentences ("*as if I were 5 years old*").

3.3.3.3 Family status

We consider family status as a binary predicate indicating whether a person is *single* or has a *partner*. Similar to labeling gender, we relied on natural language patterns containing indicative words, which are detailed in Tables 3.5 and 3.6, respectively. We distinguished two cases of indicative words: *(i) self-status indicator*, used when the speaker refers to her own status (*"I am divorced"*); and *(ii) partner indicator*, when the speaker refers to the existence of a partner (*"My boyfriend"*).

attribute	value	word/phrase indicators
gender	female	*woman*, *female*, *girl*, *lady*, *wife*, *mother*, *sister*
	male	*man*, *male*, *boy*, *husband*, *father*, *brother*
family status	single	`self-status`: *single*, *divorced*, *widow*, *spouseless*, *celibate*, *unmarried*, *unwed*, *fancy-free*
	partner	`self-status`: *married*, *engaged*, *dating* `partner`: *boyfriend*, *spouse*, *girlfriend*, *fiancee*, *lover*, *partner*, *wife*, *husband*
hobby	-	*my hobby is*, *I am/I'm fond of*, *I am/I'm keen on*, *I like*, *I enjoy*, *I go in for*, *I take joy in*, *I adore*, *I love*, *I play*, *I fancy*, *I am/I'm a fan of*, *I am/I'm fascinated by*, *I am/I'm interested in*, *I appreciate*, *I practise*, *I am/I'm mad about*

Table 3.6: Words and phrases considered as indicators used in patterns for labeling personal attributes.

We additionally collected matches of negated patterns of both *(i)* and *(ii)* in order to expand the labelled data. Furthermore, given that the indicator word *single* is often used in a more general context (e.g., '*single player*', '*single bed*'), we restricted the patterns containing this particular word, so that it should be immediately followed by punctuation, conjunctions or few allowed words like '*father*'.

3.3.3.4 Profession

To obtain profession labels we consulted a list of occupation names from Wikipedia[3] and recursively added all titles under subcategories. The resulting list consists of about 1K professions and contains a lot of fine grained occupations, some of which are redundant or ambiguous. Our strategy is to capture as many profession assertions as possible, giving the users of RedDust the opportunity to filter and group the professions depending on their specific use cases.

Each profession in the list was considered as `profession name` in the pattern `(I am|I'm) a <profession name>` that we used to label Reddit users with the *profession* attribute. After performing pattern matching against the whole Reddit dataset, we were left with 832 unique profession names in RedDust.

3.3.3.5 Hobby

Similar to collecting names of professions, we obtained a list of hobbies from Wikipedia[4] and utilized them as `hobby name` in our natural language patterns for the *hobby* attribute. We used a diverse set of patterns of the form `<phrase indicator> <hobby name>`, where `phrase indicator` is a phrase like '*my hobby is*' or '*I enjoy*', as listed in Table 3.6. Using the pattern matching approach, users in RedDust were labeled with 336 unique hobby names in total.

[3] https://en.wikipedia.org/wiki/Category:Lists_of_occupations
[4] https://en.wikipedia.org/wiki/List_of_hobbies

attribute	precision	#false positives	#disagreements
gender	0.96	2	2
age	1.0	0	2
family status	0.86	7	8
profession	0.96	2	2
hobby	0.94	3	9
avg/total	0.94	14	23

Table 3.7: Number of false positives and inter-rater agreement on RedDust.

3.3.3.6 Labeling evaluation

To validate the high-precision nature of our labeling approach, we asked three human annotators to verify the correctness of labels for each predicate. We randomly sampled 50 labeled posts for each attribute and asked annotators to indicate whether the given label matched the user's actual assertion. The decision to accept or reject the label was based on a majority vote from the annotators.

The results of this human evaluation are shown in Table 3.7. In total there were 23 instances without perfect annotator agreement (out of 250 total instances for five attributes), which indicated 14 false positives after taking a majority vote. Half of these false positives came from the family status attribute, due to ambiguous usage of words like '*partner*' in statements like "*I have a partner in this crime*". Despite such false positives, the average labeling precision for all personal attributes in RedDust is 94%. Furthermore, we also measured annotator agreement with Fleiss' kappa as 0.67 on average for all attributes, which indicates a substantial agreement; the worst agreement (0.59) was reached for the *family status* attribute.

3.3.4 Data statistics and analysis

In this section we present the quantitative and qualitative analysis of the RedDust resource. In Table 3.8 we present the overall statistics of the dataset. Figure 3.3 shows the chart of the user count per each post count. From this plot we conclude that the users in our dataset tend to have a small number of posts.

attribute	#users	#posts	#subreddits
gender	54.88K	2.49M	28.25K
age	122.20K	5.80M	44.07K
family status	11.77K	0.56M	14.76K
profession	74.86K	3.63M	37.49K
hobby	89.07K	4.42M	41.31K
total	352.78K	16.9M	165.88K

Table 3.8: Overall RedDust statistics for each attribute.

Almost 19K users in RedDust have two personal attributes known, 980 users have three

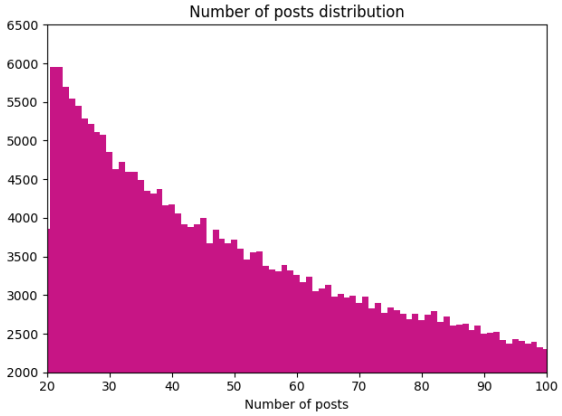


Figure 3.3: Counts of users having x number of Reddit posts.

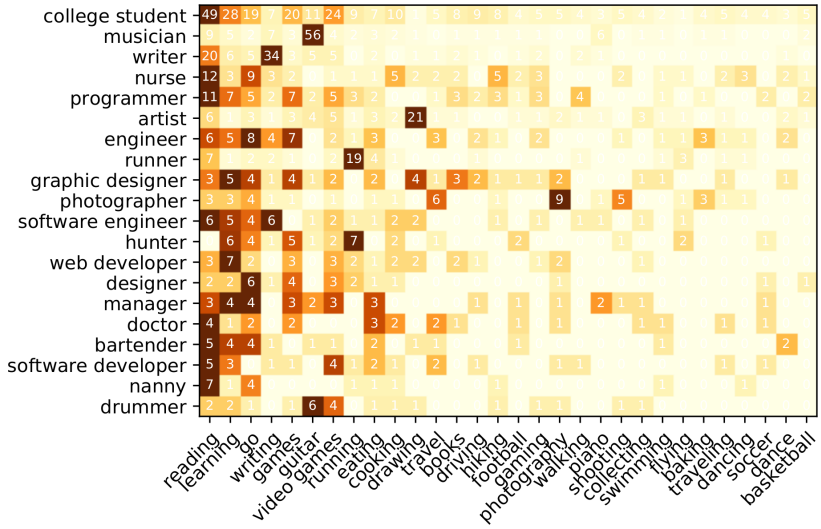


Figure 3.4: Co-occurrence of the most common professions and hobbies.

and 28 have four attributes known, which amounts to 6% of the users having multiple personal attributes in total. For such users it is interesting to look at the interplay between different personal traits, for instance, the correlation between users' occupations and general interests. In Figure 3.4 we plot a heat map which represents the co-occurring values for these two predicates. For this experiment as well as the subsequent ones, we limit the number of professions and hobbies to the top k ones ($k = 20$ and $k = 30$ for *profession* and *hobby*, respectively), sorted by the number of labeled users per value.

We observed intuitive correlations such as: *musicians* often play *guitar*; *runners* have *running* as the main interest; *college students* like to *read* but are also interested in *video games* five times as much as any other professions; and curiously, *shooting* is popular among *photographers*, most probably because of *shooting* being an ambiguous term.

We also considered other pairs of attributes, namely *profession* and *gender*, for which we show the gender distribution of each profession in Figure 3.5. The analysis revealed common

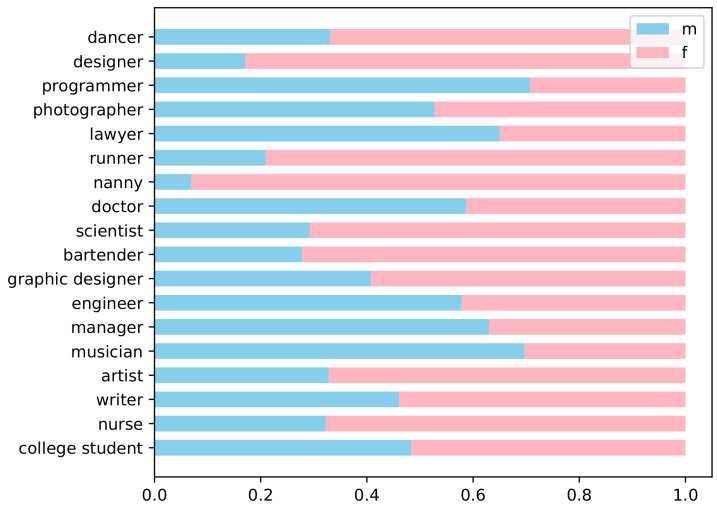


Figure 3.5: Gender distribution among professions.

prejudices like *female nannies* or *male programmers*, as well as several surprising insights (prevalence of *female runners* and *bartenders*) possibly specific to Reddit communities.

3.3.5 Labeling Reddit data with weak supervision

While using explicit statements in brackets (e.g. [35f] indicating a *35-year-old female*) and flairs is a reliable way to get *age* and *gender* of the users, utilizing natural language patterns (e.g. *"I am a doctor"*) is a much nosier method, producing many false positive errors. In addition to that, the explicit assertions, required for the pattern-based approach are rare, because the users usually hint to their personal traits with subtler cues (for example, a profession *doctor* can be deduced from the frequent use of medical terms in the posts).

Keeping in mind these limitations of the pattern-based approach we turn to the *weak supervision* method to create a refined dataset for *hobby* and *profession* user traits.

Snorkel. We used the Snorkel framework [133], which allows data labeling using weak supervision. Snorkel does not require manual evaluation of each data sample, instead it relies on the outputs of multiple *labeling functions*, such as patterns or heuristics, which are manually specified and can be potentially noisy. The accuracies and correlations of the labeling functions are then estimated by automatically deriving the *generative model* over the labeling functions. The generative model weights and combines the outputs of labeling functions to produce final list of probabilistic labels.

We modified the criteria for including Reddit submissions, so that they are: *(1)* authored by users having 10-50 posts, *(2)* 10-40 words long, and *(3)* containing a personal pronoun (except for 3rd person ones). Requirements *(1)* and *(2)* were derived from observing the word and post distributions on the full dataset. Requirement *(3)* comes from the assumption that posts containing personal pronouns are most likely to contain personal assertions. These restrictions allow us to select posts that look more similar to the real conversation (i.e., relatively short and containing references to the speakers with personal pronouns). In

profession		*hobby*		
positive	**negative**	**positive**		**negative**
i am/i'm a(n)	(no/not/don't	i am/i'm obsessed with	i am/i'm fascinated by	i hate
my profession is	within pos. patterns)	i am/i'm fond of	i am/i'm interested in	i dislike
i work as		i am/i'm keen on	i fancy	i detest
my job is		i like	i am/i'm mad about	i can't stand
my occupation is		i enjoy	i practise	(never/not/don't
i regret becoming a(n)		i love	i am/i'm into	within pos. patterns)
		i play	i am/i'm sucker for	
		i take joy in	my interest is	
		i adore	my hobby is	
		i appreciate	my passion is	
		i am/i'm fan of	my obsession is	

Table 3.9: Positive and negative patterns used in the labeling function LF1 of the Snorkel labeling model. Each pattern must be followed by possible attribute values within a context window of 2 terms.

addition, we did not consider the following subreddit types: *(i) dating*, which may provide plenty of personal information but no real conversation to infer from, and *(ii) fantasy/video games* (for *profession* attribute), because users may refer to gaming personalities. We selected only the users whose utterances contain at least one mention of the attribute values from the Wikipedia lists of professions and hobbies, resulting in around 250K and 500K candidate users for *profession* and *hobby*, respectively.

We then input the collected users into the Snorkel framework. Given a user's utterance set U, an attribute a and a possible attribute value v, Snorkel will decide on a *positive/negative* label – denoting the user as having/not having a personal trait $a : v$; or if the decision cannot be made – an *abstain* label.

We have separate labeling models for each attribute a and two labeling functions which consider: *(LF1)* the existence of the *attribute-specific patterns*, and *(LF2)* the weighted count of the words belonging to the *value-specific lexicon*.

LF1: Attribute-specific patterns. We compiled a list of positive and negative patterns for each attribute (see Table 3.9), e.g., `my hobby is <hobby-value>` vs `I hate <hobby-value>` as positive vs negative patterns for hobby. LF1 labels a user with a *positive/negative* label for each attribute value v if there exist at least one positive/negative pattern in the user's utterances U, and *abstain* label otherwise.

LF2: Value-specific lexicon. For each attribute-value pair, we used Empath [36] –pre-trained on the Reddit corpus– to build a lexicon of *typical words* (e.g., '*cider*' and '*yeast*' for *hobby:brewing*). Given seed words, Empath builds lexical categories by means of an embedding model. As our value-specific lexicon, we took the union of Empath terms for a specific attribute value and all its synonyms; each typical word is weighted by embedding similarity to the seed words. Given a user's utterance set U and an attribute value v, LF2 yields a *positive* label if the weighted count of typical words of v is above an empirically-chosen threshold, and *abstain* label otherwise.

Given a pair of user's utterance set U and a possible attribute value v, the Snorkel

probabilistic labeling model utilizes our labeling functions to predict a confidence score for the *positive* label, i.e., the user is labeled with attribute value v. As our labeled dataset, we took only the user-value pairs with confidence scores above a specific threshold.

To determine the threshold of confidence scores, we manually annotated a held-out validation set containing 100 users per attribute. Given a post and a set of attribute values mentioned explicitly in the post, the annotators had to identify whether the candidate user traits truly hold. For instance, from *"My dad bought me a **chess** board even though I enjoy **video games** more"*, *hobby:video games* is correct while *hobby:chess* is not applicable. The final annotation for each post consists of attribute values agreed by at least 2 out of 3 judges. The selected confidence threshold corresponds to the 0.9 precision of the model on the validation set. After thresholding, we obtained 13.5k users labeled with profession values and 11.7k users with hobby values.

To demonstrate that Snorkel provides the same level of quality as crowdsourcing, we calculated the precision of human annotators on the same validation set by comparing the labels of each annotator against the agreement labels. The obtained precision scores were 0.91 for profession and 0.88 for hobby, demonstrating that Snorkel is a reasonable alternative to crowdsourcing.

3.3.6 Discussion

Automatically labeling social media posts is an efficient and low-cost way to collect labeled conversations at scale. However, this approach only works for specific attributes (e.g. it is infeasible to collect *relationships* among Reddit users, because most of them are strangers to each other). Another drawback of using social media platforms is the skewed user demographics distribution, such as prevalence of young people or several professions being underrepresented. Moreover, the labels obtained from pattern search are much noisier than the crowdsourced ones, requiring further manual revision steps. Finally, the attribute value lists automatically collected from Wikipedia can be further refined by merging redundant values and adding the missing ones.

Chapter 4

Hidden Attribute Models

Contents

Open-domain dialogue agents must be able to converse about many topics while incorporating knowledge about the user into the conversation. The background information about the user's demographics, such as *age* or *gender*, can help the chat-bot adjust its conversational style, make relevant recommendations and initiate engaging discussions. Instead of asking the users to manually provide their personal information or seek it in the external sources, we propose to directly extract such facts from the user's dialogues. This problem is more challenging than the established task of information extraction from scientific publications or Wikipedia articles, because dialogues often give merely implicit cues about the speaker.

We propose methods for inferring personal attributes, such as *profession*, *age*, *gender* and *family status*, from conversations using deep learning. Specifically, we propose several *Hidden Attribute Models*, which are neural networks leveraging attention mechanisms and

embeddings. Our methods are trained on a per-predicate basis to output rankings of object values for a given subject-predicate combination (e.g., ranking *doctor* and *nurse* professions high when speakers talk about patients, emergency rooms, etc). Experiments with various conversational texts including Reddit discussions, movie scripts and a collection of crowdsourced personal dialogues demonstrate that our methods outperform state-of-the-art baselines, providing accurate predictions of personal attributes.

4.1 Introduction

Motivation: While interest in dialogue agents has grown rapidly in recent years, creating agents capable of holding personalized conversations remains a challenge. The knowledge of the user's demographic attributes, such as *age* or *family status*, is a crucial step towards a user-friendly dialogue system, capable of adjusting its speech style and offering relevant suggestions with respect to the user's traits.

For meaningful and diverse dialogues with a real person, a system should be able to infer knowledge about the person's background from her utterances. Consider the following example, where H stands for a human and A for a dialogue agent:

H: *What's the best place for having brekky?*
A: *The porridge at Bread and Cocoa is great.*
H: *Any suggestions for us and the kids later? We already visited the zoo.*
A: *There's the San Francisco Dungeon, an amusement ride with scary city history.*

From the word 'brekky' in the first H utterance, the system understands that the user is Australian and may thus like porridge for breakfast. However, the cue is missed that the user is with pre-teen children (talking about kids and the zoo), and the resulting suggestion is inappropriate for young children. Instead, with awareness of this knowledge, a better reply could have been:

A: *I bet the kids loved the sea lions, so you should also see the dolphins at Aquarium of the Bay*

A possible remedy to improve this situation is to include user information into an end-to-end learning system for the dialogue agent. However, any user information would be bound to latent representations rather than explicit attributes. Instead, we propose to capture such attributes explicitly and add them to a personal knowledge base, which will then be a distant source of background knowledge for personalization in downstream applications such as Web-based chatbots and agents in online forums.

To populate the PKB without the user's manual supervision, we need to leverage the methods for automatic personal information extraction. There has been ample work on information extraction from structured external sources, such as Wikipedia entries or news stories. However, there is little hope of finding the personal information about each individual user in encyclopedic articles.

Instead, personal facts can be extracted from unstructured textual sources, such as user's conversational data. Such data, in form of dialogue transcriptions or social media posts, is

abundant and rich in signals about the given user's persona. However, conventional methods for information extraction from well-comprehensible text genres fail to perform properly given conversations as input. Dialogue utterances are short, noisy and colloquial, which necessitates creating novel extraction methods, tailored specifically to conversational data.

This work addresses these issues by proposing methods to *infer* personal facts from dialogues based on implicit cues. The proposed approach takes advantage of the hierarchical dialogue structure to outperform previous information extraction models.

State of the Art and its Limitations: Currently the most successful dialogue agents are task-oriented, for instance, supporting users with car navigation or delivery orders (e.g., [99, 109]). This makes the task considerably easier, because the system has to focus on specific words, related to the topic. We, in contrast, strive to be able to extract information for domain-independent dialogue, from everyday conversation, where the relevant facts are only latent. General-purpose chatbot agents show decent performance in benchmarks (e.g., [44, 83, 148]), but critically rely on sufficient training data and tend to lack robustness when users behave in highly varying ways. Very few approaches have considered incorporating explicit knowledge on individual users, and these approaches have assumed that personal attributes are explicitly mentioned in the text [60, 85, 183].

To illustrate that identifying explicit mentions of attributes is insufficient, we developed an oracle to obtain an upper bound on the performance of pattern-based approaches, such as [85]. This oracle, which is described in Section 4.5.2, assumes that we have perfect pattern matching that correctly extracts an attribute value every time it is mentioned. This oracle routinely performs substantially worse than our proposed methods, demonstrating that extracting information from utterances requires inferring the presence of attribute values that are never explicitly stated.

On the other hand, many efforts have considered the problem of profiling social media users in order to predict latent attributes such as *age*, *gender*, or *regional origin* (e.g., [10, 14, 34, 40, 70, 130, 138, 140, 159]). While social media posts and utterances are similar in that both are informal, the former can be associated with many non-textual features that are unavailable outside of the social media domain (e.g., social-network friends, likes, etc. and explicit self-portraits of users). We consider several user profiling baselines that rely on only textual features and find that they do not perform well on our task of inferring attributes from conversational utterances.

Approach and Contributions: We devise a neural architecture, called **H**idden **A**ttribute **M**odels (HAMs) [152], trained with subject-predicate-object triples to predict objects on a per-predicate basis, e.g., for a subject's *profession* or *family status*. The underlying neural network learns to predict a scoring of different objects (e.g., different professions) for a given subject-predicate pair by using attention within and across utterances to infer object values. For example, as illustrated later in Table 4.7, our approach infers that a subject who often uses terms like '*theory*', '*mathematical*', and '*species*' is likely to be a *scientist*, while a subject who uses terms like '*senate*', '*reporters*', and '*president*' may be a *politician*.

The salient contributions of this work are the following:

- a viable method for learning personal attributes from conversations, based on neural networks with novel ways of leveraging attention mechanisms and embeddings,
- an extensive experimental evaluation of various methods on Reddit, movie script dialogues and crowdsourced personalized conversations (PersonaChat),
- an experimental evaluation of the transfer learning approach: leveraging ample data from user-generated social media texts (Reddit) for inferring users' latent attributes from data-scarce speech-based dialogues (movie scripts and PersonaChat).

4.2 Related work

In this section we discuss related work concerning the methods that are used in HAMs. First, we give an overview of neural architectures utilizing attention mechanism, which is the main building block in our best performing models. Second, we describe the approaches for building hierarchical representations of the conversational data. We also refer the reader to Section 2.1 for a comprehensive overview of the author profiling methods.

4.2.1 Neural Models with Attention

The role of attention weights has been studied for various neural models, including feed-forward networks [158], CNNs, [177] and RNNs [6]. Recently, neural models enhanced with attention mechanisms have boosted the results on various NLP tasks [149, 174, 188], particularly in conversational domain for response generation [2, 180] or spoken language understanding [21].

In response generation task, attention is used to align the context and target utterance representations [2]. Zhang et al. [180] extend it with additional self-attention layers for both context and response representations. Chen et al. [21] uses attention to estimate the relevance of the previous knowledge stored in memory to the input utterances; the response is produced using the attention distribution, calculated by matching each input utterance to the memory vectors.

Transformer [158] is a state-of-the-art sequence-to-sequence deep learning model based on self-attention mechanism. Transformer is used across various NLP tasks, both as a standalone model and as a part of other neural architectures. In particular, in conversation domain Transformer has been used to produce context-aware utterance representations [84, 144].

4.2.2 Hierarchical conversational models

Hierarchical models to represent conversations were introduced by Serban et al. [143], who applied RNNs to hierarchically build the representations of utterances and the dialogue context, solving response generation task. Xing et al. [172] also decoded conversational responses, introducing attention mechanism into the hierarchical encoder architecture. In Xing et al. [172] the utterance and word representations are formed as the attention-weighted averages of the hidden states in the word and utterance level RNNs.

Hierarchical attention models are also utilized for other conversational NLP tasks, such as dialogue state tracking [144] or emotion recognition in conversations [84, 98]. A common approach is to create word representations with BERT and utterance representations with Transformer encoder [84, 144] or RNN [98].

There is also ample research on applying hierarchical attention to speaker attribute prediction. Lynn et al. [97] use attention mechanism with the word and utterance representations created by an RNN to predict personality traits of the Facebook users. The study [86] exploits hierarchical model to predict *age*, *gender* and *location* information of Weibo users.

Compared to most hierarchical models, the architecture of HAMs is more light-weight, because it creates speaker representations with an attention mechanism directly, without additionally running an RNN or Transformer models on the attention-weighted words. Thus, HAMs are less prone to overfitting and require less computational resources. Regardless of its simplicity, our proposed architecture can still make meaningful predictions in classification tasks with large number of classes, such as *profession* prediction, as opposed to few possible classes in related studies [86, 97].

4.3 Methodology

In this section we describe *Hidden Attribute Models* (HAMs) for predicting the values of a given personal attribute using a sequence of utterances made by a speaker. Formally, given a speaker S and an attribute P, our goal is to predict a probability distribution over attribute values O for the attribute, based on the speaker's utterances from a dialogue corpus (e.g., a movie script). Each speaker S is associated with a sequence of N utterances $[U_1, U_2, ..., U_N]$ containing M terms each, $U_1 = [U_{1,1}, U_{1,2}, ..., U_{1,M}]$. Each term $U_{n,m}$ is represented as a d-dimensional word embedding.

HAMs can be described in terms of three functions and their outputs:

1. f_{utter} creates a representation R_n^{utter} of the nth utterance given the terms in the utterance:
$$R_n^{utter} = f_{utter}(U_{n,1}, U_{n,2}, ..., U_{n,M}) \tag{4.1}$$

2. f_{subj} creates a speaker representation R^{subj} given the sequence of utterance representations:
$$R^{sp} = f_{subj}(R_1^{utter}, R_2^{utter}, ..., R_N^{utter}) \tag{4.2}$$

3. f_{obj} outputs a probability distribution over attribute values O given the speaker representation:
$$O = f_{obj}(R^{subj}) \tag{4.3}$$

   Depending on the attribute which value is being predicted, this distribution is used to either make a prediction (for binary attributes, e.g. *gender*) or to produce a ranked list of object values (for multi-class attributes, such as *profession*).

In the following we describe Hidden Attribute Models by instantiating these functions.

HAM avg illustrates the most straightforward way to combine word and utterance representations. In this model,

$$avg(X) = \sum_{i=1}^{|X|} X_i \tag{4.4}$$

serves as both f_{utter} and f_{sp}; the n-th utterance representation R_n^{utter} is created by averaging the terms in the n-th utterance and the speakert representation R^{sp} is created by averaging the N utterance representations together. Two stacked fully connected layers serve as the function f_{obj},

$$FC(x) = \sigma(Wx + b) \tag{4.5}$$

where σ is an activation function and W and b are learned weights. The full HAM$_{\text{avg}}$ model is then

$$R_n^{utter} = avg(U_n) \tag{4.6}$$

$$R^{subj} = avg(R^{utter}) \tag{4.7}$$

$$O = FC_1(FC_2(R^{subj})) \tag{4.8}$$

where FC_2 uses a sigmoid activation and FC_1 uses a softmax activation function in order to predict a probability distribution over object values.

HAM 2attn extends HAM$_{\text{avg}}$ with two self-attention mechanisms, allowing the model to learn which terms and utterances to focus on for the given predicate. In this model the utterance representations and speaker representations are computed using attention-weighted averages,

$$attn\text{-}avg(X, \alpha) = \sum_{i=1}^{|X|} X_i \alpha_i \tag{4.9}$$

with the attention weights calculated over utterance terms and utterance representations, respectively. That is, $f_{utter}(X) = attn\text{-}avg(X, \alpha^{term})$ and $f_{so}(X) = attn\text{-}avg(X, \alpha^{utter})$, where the attention weights for each term in an utterance U_i are calculated as

$$w_i^{term} = \sigma(W^{term} U_i + b^{term}) \tag{4.10}$$

$$\alpha_{i,j}^{term} = \frac{exp(w_{i,j}^{term})}{\sum_j exp(w_{i,j}^{term})} \tag{4.11}$$

and the utterance representation weights α^{utter} are calculated analogously over R^{utter}. Given these attention weights, the HAM$_{\text{2attn}}$ model is

$$R_n^{utter} = attn\text{-}avg(U_n, \alpha^{term}) \tag{4.12}$$

$$R^{sp} = attn\text{-}avg(R^{utter}, \alpha^{utter}) \tag{4.13}$$

$$O = FC(R^{sp}) \tag{4.14}$$

where f_{obj} function FC uses a softmax activation function as in the previous model.

HAM$_{\text{CNN}}$ considers n-grams when building utterance representations, unlike both previous models that treat each utterance as a bag of words. In this model f_{utter} is implemented with a text classification CNN [71] with a ReLU activation function and k-max pooling across utterance terms (i.e., each filter's top k values are kept). A second k-max pooling operation across utterance representations serves as f_{sp}. As in the previous model, a single fully connected layer with a softmax activation function serves as f_{obj}.

HAM$_{\text{CNN-attn}}$ extends HAM$_{\text{CNN}}$ by using attention to combine utterance representations into the speaker representation. This mirrors the approach used by HAM$_{\text{2attn}}$, with $f_{sp} = attn\text{-}avg(X, \alpha^{utter})$ and α^{utter} computed using Equations 4.10 and 4.11 as before. This model uses the same f_{utter} and f_{obj} as HAM$_{\text{CNN}}$. That is, utterance representations are produced using a CNN with k-max pooling, and a single fully connected layer produces the model's output.

Training All HAMs were trained with gradient descent to minimize a cross-entropy loss. We use the Adam optimizer [72] with its default values and apply an L2 weight decay (2e-7) to the loss.

4.4 Data acquisition and processing

In experiments with HAMs we used three different datasets, reflecting various aspects of conversational data: *(i)* movie scripts (*MovieChAtt* dataset, described in Chapter 3.2.2); *(ii)* social media submissions (a subset of the *RedDust* dataset, described in Chapter 3.3); and *(iii)* artificially created dialogues (PersonaChat dataset [183]). In this section we provide details on these datasets.

MovieChAtt dataset. We use the dataset of the movie characters' utterances, described in Section 3.2.2, annotated with *profession*, *age* and *gender* attributes. In summary, we obtained 1,963 characters labeled with *gender* (*male* or *female*), 4,548 characters labeled with *age* (classified into one of the bins from *child*, *teenager*, *adult*, *middle-aged* and *senior*) and 1,405 characters labeled with *profession* (out of 43 profession values, given in Table 3.1).

For each character in the annotated set we extracted the sequence of their utterances in the movie. Each utterance is represented as a sequence of words, excluding stop words, the 1,000 most common first names[1], and words that occur in fewer than four different movies. The latter two types of words are excluded in order to prevent the model from relying on movie-specific or character-specific signals that will not generalize.

RedDust dataset. For experiments with HAMs we used a part of the RedDust dataset, covering all four considered predicates. Specifically, we tapped into two subforums on Reddit: "*iama*", where anyone can ask questions to a particular person, and "*askreddit*", with more general conversations. In selecting these subforums we followed two criteria: *(1)* they are not concerned with fictional topics (e.g. computer games) and *(2)* they are not too topic-specific, as this could heavily bias the classification of user attributes.

[1]Removed to prevent overfitting, http://catalog.data.gov/dataset/baby-names-from-social-security-card-applications-national-level-data

As we discussed in Chapter 3.3, the users in the RedDust dataset were labeled using three techniques: natural language patterns, flairs (short description attached to a username) and bracketed assertions (for example, '[34f]' indicating 34 *year old female*). For expleriments with HAMs we used only a subset of users which were labeled with natural language patterns. Flairs in the selected two subreddits do not indicate the users' genders; the bracketed assertions, which are mostly featured in dating subreddits, are rare in "*iama*" and "*askreddit*".

We removed users who claim multiple values for the same attribute, as we allow only for single-label classification. To further increase data quality, we rated users by the language style, to give preference to those whose posts sound more like the utterances in a dialogue. This rating was computed as the fraction of the user's posts that contain personal pronouns, because pronouns are known to be abundant in the dialogue data.

The test set, disjoint from training data was created in the same manner and further checked by manual annotators, considering that the above mentioned patterns may also produce false positives. For example, the patterns can indicate a wrong profession from utterances such as "*they think I am a doctor*" or "*I dreamed I am an astronout*"; or wrong age and family status from "*I am 10 years old boy's mother*" or "*I am a single child*". The final RedDust test set consists of approximately 400 users per predicate.

PersonaChat dataset. We also explore the robustness of our models using the PersonaChat corpus [183], which consists of conversations collected via MTurk. The workers were given 5-sentence-long persona descriptions (e.g., "*I am an artist*", "*I like to build model spaceships*") and asked to incorporate these facts into a short conversation (up to 8 turns) with another worker. We split these conversations by persona, yielding a sequence of 3 or 4 utterances for each persona in a conversation.

We automatically labeled personas with *profession* and *gender* attributes by looking for patterns `I am/I'm a(n) <term>` in persona descriptions, where `<term>` is either a profession label or a gender-indicating noun ('*woman*', '*uncle*', '*mother*', etc). We manually labeled persons with *family status* by identifying persona descriptions containing related words ('*single*', '*married*', '*lover*', etc) and labeling the corresponding persona as *single* or *not single*. Overall, we collected 1,147 personas labeled with *profession*, 1,316 with *gender*, and 2,302 labeled with *family status*.

Limitations. The number of predicates we consider for each dataset are limited by the nature of these datasets. We do not consider the *family status* predicate for MovieChAtt, because the necessary information is often not easily available from the Wikipedia articles. Similarly, we do not consider the *age* predicate on the PersonaChat dataset, because this attribute is not easily found in the persona descriptions. More generally, all users in our datasets are labeled with exactly one attribute value for each predicate. In a real setting it may be impossible to infer any attribute value for some users, whereas other users may have multiple correct values.

4.5 Experimental setup

4.5.1 Data

We randomly split the MovieChAtt and PersonaChat datasets into training (90%) and testing (10%) sets. We tuned models' hyperparameters by performing a grid search with 10-fold cross validation on the training set.

For the binary attributes *family status* and *gender*, we balanced the number of speakers in each class. For the multi-valued *profession* and *age* attributes, which have very skewed value distributions, we did not balance the number of speakers in the test set. During training we performed downsampling to reduce the imbalance; each batch consisted of an equal amount (set to 3) of training samples per class, and these samples were drawn randomly for each batch. This both removes the class imbalance in training data and ensures that ultimately the model sees all instances during training, regardless of the class size.

4.5.2 Baselines

Pattern matching oracle. Assuming that we have a perfect sequence tagging model (e.g., similar to an approach used by Li et al. [85]) that extracts a correct attribute value every time one appears in an utterance, we can determine the upper-bound performance of such sequence tagging approaches. Note that this type of model assumes that attribute values explicitly appear in the text. In order to avoid vocabulary mismatches between our attribute value lists and the attribute values explicitly mentioned in the data, we augment our attribute values with synonyms identified in the data (e.g., we add terms like '*soldier*' and '*sergeant*' as synonyms of the the *profession* attribute value *military personnel*). For MRR and accuracy metrics calculation, a speaker receives a score of 1 if the correct attribute value appears in any one of a speaker's utterances (when multiple attribute values are mentioned, we assume the oracle picks the correct one). If the correct attribute value never appears, we assume the model returns a random ordering of attribute values and use the expectation over this list (i.e., given $|V|$ attribute values, the speaker receives a score of $\frac{1}{0.5|V|}$ for MRR and $\frac{1}{|V|}$ for accuracy). This oracle method does not provide class confidence scores, so we do not report AUROC with this method.

Embedding similarity. Given an utterance representation created by averaging the embeddings of the words within the utterance, we compute the cosine similarity between this representation and the embeddings of each attribute value. In this and the following baselines we used 300-dimensional word2vec embeddings pre-trained on the *Google News* corpus [108] to represent the terms in the utterances.

Logistic regression. Given an averaged utterance representation (as used with embedding similarity), we apply a multinomial logistic regression model [104] to classify the representation into one of the possible attribute values. This model obtains a ranking of the attribute values by ordering the per-value probabilities of its output.

Multilayer Perceptron (MLP). Given an averaged utterance representation (as used with embedding similarity), we apply an MLP with one hidden layer of size 100 to classify

the utterance representation as one of the possible attribute values. Similarly to the previous model, MLP can be used to obtain attribute values ranking by considering the per-value probabilities.

The embedding similarity, logistic regression, and MLP baselines are distantly supervised, because the speaker's labels are applied to each of the speaker's utterances. While this is necessary because the baselines do not incorporate the notion of a speaker with multiple utterances, it results in noisy labels because it is unlikely that every utterance will contain information about the speaker's attributes. We address this issue by using a window of $k = 4$ (determined by a grid search) concatenated utterances as input to each of these methods. With these distantly supervised models, the label prediction scores are summed across all utterances for a single speaker and then ranked.

CNN [10]. We consider the Convolutional Neural Network (CNN) architecture proposed by Bayot and Gonçalves for the task of predicting the *age* and *gender* of Twitter users. This approach is a simpler variant of HAM_{CNN} in which f_{utter} is implemented with a *tanh* activation function and max pooling (i.e., $k = 1$) and f_{obj} is a fully connected layer with dropout ($p = 0.5$) and a softmax activation function. The CNN is applied to individual utterances and the user obtains a label by taking the majority vote accross the utterances, which differs from the in-model aggregation performed by HAM_{CNN}.

New Groningen Author-profiling Model (N-GrAM) [9]. Following the best performing system at CLEF 2017's PAN shared task on *author profiling* [41], we implemented a classification model using a linear Support Vector Machine (SVM) [26] that utilizes the following features: character n-grams with $n = 3, 4, 5$, and term unigrams and bigrams with sublinear TF-IDF weighting.

Neural Clusters (W2V-C). We consider the best classification model reported by Preoţiuc-Pietro et al. [123] for predicting the *occupational class* of Twitter users, which is a Gaussian Process (GP) classifier [22] with *neural clusters* (W2V-C) as features. Neural clusters were obtained by applying spectral clustering on a word similarity matrix (via cosine similarity of pre-trained word embeddings) to obtain $n = 200$ word clusters. Each post's feature vector is then represented as the ratio of words from each cluster.

For both N-GrAM and W2V-C baselines, flattened representations of the speaker's utterances are used. That is, the model's input is a concatenation of all of a given user's utterances.

4.5.3 Hyperparameters

Hyperparameters were chosen by grid search using ten-fold cross validation on the training set. Due to the limited amount of data, we found that a minibatch size of 4 users performed best on MovieChAtt and PersonaChat. All models were trained with a minibatch size of 32 on the RedDust dataset. Similar to the baselines, we instantiated words' representations with word2vec embeddings pre-trained on the Google News corpus. We set the number of utterances per character $N = 40$ and the number of terms per utterance $M = 40$, and truncate or zero pad the sequences as needed.

- **HAM$_{\text{avg}}$** uses a hidden layer of size 100 with the sigmoid activation function. The model was trained for 30 epochs.
- **HAM$_{\text{CNN}}$** uses 178 kernels of size 2 and k-max pooling with $k = 5$. The model was trained for 40 epochs.
- **HAM$_{\text{2attn}}$** uses a sigmoid activation with both attention layers and with the prediction layer. The model was trained for 150 epochs.
- **HAM$_{\text{CNN-attn}}$** uses 128 kernels of size 2. The model was trained for 50 epochs.

We implemented HAMs and neural baselines using PyTorch; logistic regression, SVM and Gaussian Process classifiers were implemented using Scikit-learn. The code for the models and the data are available at https://github.com/Anna146/HiddenAttributeModels.

4.5.4 Evaluation metrics

Due to the difficulty of performing classification over many attribute values, a ranking metric is more informative for the multi-valued attributes *profession* and *age category*, thus, we report MRR for these attributes. We obtain the ranking of the attribute values for a movie character or a persona by considering the models' output attribute value probabilities. We report both *macro MRR*, in which we calculate a reciprocal rank for each attribute value before averaging, and *micro MRR*, which averages across each speaker's reciprocal rank.

For binary attributes *gender* and *family status*, we report models' performance in terms of accuracy. Additionally we report micro AUROC for all attributes. For multi-valued attributes, we binarize the labels in a one-vs-all fashion.

4.6 Results and Discussion

4.6.1 Main Findings

In Tables 4.1, 4.1, 4.4 and 4.3 we report results for HAMs and the baselines on all datasets (MovieChAtt, PersonaChat and RedDust) for all considered attributes (*profession*, *gender*, *age* and *family status*). In the tables the results marked with * significantly differ from the best performing method (highlighted with bold font) with a p-value cutoff $p < 0.05$, as measured by a paired t-test (MRR) or McNemar's test (Acc and AUROC).

We do not report results for the *pattern oracle* baseline as we evaluated this baseline solely on the MovieChAtt dataset, because the oracle essentially replicates the way we labeled persona descriptions and posts in the PersonaChat and RedDust datasets, respectively. The pattern oracle baseline yields 0.21/0.20 micro/macro MRR for *profession*, 0.67 accuracy for *gender*, and 0.41/0.40 micro/macro MRR for *age*. HAMs significantly outperform this baseline, indicating that identifying explicit mentions of attribute values is insufficient in our dialogue setting.

HAMs outperform the distantly supervised models (i.e., *embedding similarity*, *logistic regression* and *Multilayer Perceptron (MLP)*) in the vast majority of cases. MLP and logistic

Models	profession					
	MovieChAtt		PersonaChat		RedDust	
	MRR micro / macro	AUROC	MRR micro / macro	AUROC	MRR micro / macro	AUROC
Embedding sim.	0.22* / 0.14*	0.60*	0.30* / 0.25*	0.63*	0.15* / 0.13*	0.59*
Logistic reg.	0.46* / 0.20*	0.76*	0.81* / 0.77*	0.58*	0.13* / 0.19*	0.57
MLP	**0.47** / 0.20	0.75	0.86* / 0.77*	0.97	0.46 / 0.23	0.78
N-GrAM [9]	0.21* / 0.16*	0.62*	0.83* / 0.83	0.88	0.17 / 0.26	0.64*
W2V-C [123]	0.25* / 0.13*	0.74*	0.59 / 0.46	0.89	0.27* / 0.17*	0.74*
CNN [10]	0.19* / 0.20*	0.66*	0.77* / 0.77*	0.81*	0.26* / 0.24*	0.76*
HAM_{avg}	0.39* / 0.37*	0.81*	0.86* / 0.91*	0.98*	0.34* / 0.22*	0.82*
HAM_{CNN}	0.42 / 0.37	0.83	**0.96** / **0.94**	**0.99**	0.36* / 0.37*	0.86*
$HAM_{CNN\text{-}attn}$	0.43 / **0.50**	**0.85**	0.90 / 0.93	**0.99**	**0.51** / 0.40	**0.9**
HAM_{2attn}	0.39 / 0.34	0.84	0.94 / 0.93	**0.99**	0.43 / **0.42**	0.89

Table 4.1: Comparison of models on all datasets for *profession* attribute.

Models	gender					
	MovieChAtt		PersonaChat		RedDust	
	Acc	AUROC	Acc	AUROC	Acc	AUROC
Embedding sim.	0.52*	0.54*	0.49	0.50	0.61*	0.60*
Logistic reg.	0.59	0.62	0.86	0.93	0.69*	0.75*
MLP	0.57*	0.60*	0.80	0.87	0.71	0.77
N-GrAM [9]	0.57	0.58	0.86	0.87	0.66*	0.71*
W2V-C [123]	0.62	0.66	0.73*	0.80*	0.64*	0.73*
CNN [10]	0.60	0.60	0.72*	0.73*	0.61*	0.61*
HAM_{avg}	0.72	0.82	0.79	0.87	**0.86**	0.92
HAM_{CNN}	0.75	**0.85**	0.95	**0.99**	**0.86**	0.93*
$HAM_{CNN\text{-}attn}$	**0.77**	0.84	**0.96**	0.97	0.85	**0.94**
HAM_{2attn}	0.69	0.77	0.94	0.98	0.80	0.91

Table 4.2: Comparison of models on all datasets for *gender* attribute.

regression perform best in several occasions for *profession* and *age* attributes when micro MRR is considered. However, their macro MRR scores fall behind HAMs', showing that HAMs are better at dealing with multi-valued attributes having skewed distribution. The low performance of these distantly supervised methods may be related to their strong assumption that every sequence of four utterances contains information about the attribute being predicted.

Comparing with baselines from prior work, HAMs significantly outperform N-GrAM in many cases, suggesting that representing utterances using word embeddings, instead of merely character and word n-grams, is important for this task. Using neural clusters (W2V-C) as features for the classification task [123] works quite well for the *age* attribute, where different 'topics' may correlate with different age categories (e.g. '*video game*' for *teenager* and '*office*' for *adult*). However, W2V-C is often significantly worse for the *profession*, *gender*, and *family status* attributes, which may be caused by similar discriminative

Models	family status			
	PersonaChat		RedDust	
	Acc	AUROC	Acc	AUROC
Embedding sim.	0.41*	0.49*	0.42*	0.47*
Logistic reg.	0.75*	0.84*	0.71	0.74
MLP	0.70	0.80	0.62*	0.60*
N-GrAM [9]	0.85	0.86	0.45*	0.47*
W2V-C [123]	0.74*	0.82*	0.70	0.78
CNN [10]	0.74	0.74	0.69	0.69
HAM$_{avg}$	0.80	0.91	0.67	0.72
HAM$_{CNN}$	**0.93**	**0.99**	0.52*	0.62*
HAM$_{CNN\text{-}attn}$	0.92	0.98	**0.70**	**0.78**
HAM$_{2attn}$	0.88	0.94	0.64	0.67

Table 4.3: Comparison of models on all datasets for *family status* attribute.

words (e.g., '*husband*'/'*wife*' for *gender*) being clustered together in the same topic. The CNN baseline [10] is significantly worse than the best method in the majority of cases. Furthermore, it generally performs substantially worse than HAM$_{CNN}$, further illustrating the advantage of aggregating utterances within the model.

In general, HAM$_{avg}$ performs worse than the other HAMs, demonstrating that simple averaging is insufficient for representing utterances and speakers. In most cases HAM$_{CNN}$ performs slightly worse than HAM$_{CNN\text{-}attn}$, demonstrating the value of exploiting the attention mechanism to combine speaker's utterances.

HAM$_{CNN\text{-}attn}$ and HAM$_{2attn}$ achieve the strongest performance across attributes, with HAM$_{CNN\text{-}attn}$ generally performing better. HAM$_{CNN\text{-}attn}$ performs particularly well on the *gender* and *family status* attributes, where detecting bigrams may yield an advantage. For example, HAM$_{2attn}$ places high attention weights on terms like '*family*' and '*girlfriend*' where the previous term may be a useful signal (e.g., '*my family*' vs. '*that family*').

The gap between the baselines and HAMs is often smaller on PersonaChat compared with the other two datasets, illustrating the simplicity of crowdsourced dialogues as compared to movie scripts or Reddit discussions. This is also supported by the fact that the maximum metrics on PersonaChat are much higher. There are several factors that may be responsible for this: *(1)* the dialogues in PersonaChat were created by the crowdworkers with the goal of using predefined personal facts, which often leads to those facts being stated in a straightforward manner (e.g., saying "*My job is a writer*" given the persona description sentence "*I am a writer*"); *(2)* PersonaChat utterances are much shorter and there are far fewer utterances per character (i.e., a maximum of 4 in PersonaChat vs. a minimum of 20 in MovieChAtt), leading to a higher density of information related to attributes; and *(3)* the same persona descriptions in PersonaChat are used across multiple separate dialogue sessions, giving models an opportunity to learn specific personas.

For the sake of brevity we neither instantiate nor report results for LSTM-based HAMs, such as $f_{utter} = LSTM$ and $f_{subj} = attn\text{-}avg$ or $f_{subj} = LSTM$. These models were unable to outperform HAM$_{avg}$, with the best variant obtaining a micro MRR of only 0.31 after grid search (profession attribute on MovieChAtt, Table 4.1). This is in line with recent results

Models	age			
	MovieChAtt		RedDust	
	MRR micro / macro	AUROC	MRR micro / macro	AUROC
Embedding sim.	0.45* / 0.45*	0.61*	0.55* / 0.44*	0.56*
Logistic reg.	0.65* / 0.49*	0.76	**0.80** / 0.61	0.87
MLP	0.64* / 0.48*	0.83	0.78 / 0.48	0.88
N-GrAM [9]	0.69 / 0.47	0.85	0.48* / 0.53*	0.55*
W2V-C [123]	0.67 / 0.45	0.86	0.75 / 0.51	0.88
CNN [10]	0.66* / 0.62*	0.83	0.68* / 0.65*	0.79*
HAM $_{avg}$	0.62* / 0.59	0.76*	0.67 / 0.67	0.77*
HAM $_{CNN}$	0.73* / **0.63**	0.84	0.73* / 0.61*	0.89*
HAM $_{CNN\text{-}attn}$	0.73 / 0.60	**0.86**	0.79 / **0.68**	**0.90**
HAM $_{2attn}$	**0.74** / 0.6	0.85	0.72 / 0.6	0.82

Table 4.4: Comparison of models on all datasets for *age* attribute.

suggesting that RNNs are not ideal for identifying semantic features [150].

4.6.2 Study on word embeddings

In Section 4.6.1 we represented terms using embeddings from a word2vec skip-gram model trained on Google News [108]. In this study we compare the Google News embeddings with word2vec embeddings trained on Reddit posts, GloVe [117] embeddings trained on Common Crawl, and GloVe embeddings trained on Twitter. We also consider *ELMo* [118], a contextualized embedding model. To capture semantic variations, this model creates a contextualized character-based representation of words using a bidirectional language model. We use AllenNLP's small ELMo model trained on the 1 Billion Word Benchmark of news crawl data from WMT 2011 [18].

Model	Corpus	HAM $_{CNN\text{-}attn}$		HAM $_{2attn}$	
		MRR micro / macro	AU-ROC	MRR micro / macro	AU-ROC
word2vec (skip-gram)	Google News	0.42 / **0.44**	0.77	0.39 / 0.37	**0.83**
	Reddit	**0.43** / 0.37	**0.82**	**0.50** / 0.37	**0.83**
GloVe	Common Crawl	0.40 / 0.37	0.76	0.40 / **0.39**	0.82
	Twitter	0.39 / 0.35	0.67	0.36 / 0.34	0.81
ELMo	WMT News	0.38 / 0.32	0.76	0.37 / 0.37	0.83

Table 4.5: Comparison of embedding models trained on different datasets for identifying *profession* attribute.

Given the higher model variance on the *profession* attribute on MovieChAtt, we restrict the study to this attribute and dataset. We evaluated the two best performing HAMs, i.e., HAM $_{CNN\text{-}attn}$ and HAM $_{2attn}$. Table 4.5 shows the results obtained with the various embedding methods trained on different corpora. The difference in performance does not greatly vary across embedding models and corpora, with Google News embeddings performing

	MRR micro / macro	AUROC
HAM $_{2attn}$	**0.57 / 0.42**	**0.84**
− attention on terms	0.49 / 0.40	0.81
− attention on R^{utter}	0.48 / 0.34	0.82

Table 4.6: Ablation study for the *profession* attribute.

best in terms of macro MRR and Reddit embeddings performing best in terms of micro MRR. Despite their strong performance on some NLP tasks, the ELMo contextualized embeddings do not yield a performance boost for any method or metric. We view this observation as an indicator that the choice of term embedding method is not very significant for this task compared to the method used to combine terms into an utterance representation.

4.6.3 Ablation study

We performed an ablation study in order to determine the performance impact of the HAMs' components. As in the previous section, we restrict this study to the inference of the *profession* attribute on MovieChAtt dataset. Ablation results for HAM $_{2attn}$ using cross validation on the training set are shown in Table 5.4. Replacing either representation function (i.e., f_{utter} or f_{subj}) with an averaging operation reduces performance, as shown in the last two lines. Attention on utterance representations (R^{utter}) is slightly more important in terms of MRR, but both types of attention contribute to HAM $_{2attn}$'s performance. Similarly, removing both types of attention corresponds to HAM $_{avg}$, which consistently underperforms HAM $_{2attn}$ in Tables 4.1, 4.2, 4.4 and 4.3.

Removing attention from HAM $_{CNN\text{-}attn}$ yields HAM $_{CNN}$, which consistently performs worse than HAM $_{CNN\text{-}attn}$ in Tables 4.1, 4.2, 4.4 and 4.3, supporting the observation that attention is important for performance on our task. Intuitively, attention provides a useful signal because it allows the model to focus on only the terms containing important information about an attribute.

0.065 this clown almost blew mission security on the street. i'm not jumping with him.
0.062 american. come to get you out. can you walk? what's your name? colonel.
0.066 how you doing brewer? guard barracks. take some shots. break your leg i'll have to shoot you.
0.034 you comin'? can you handle the door gun? brewer! you know what that thing's packing? forget it.

(a) *profession*: military personnel

0.098 am i really gonna have my own room? yaay! but what if he dies and has to go to the pet sematary? i want to fly it! can i fly it now mommy!
0.084 will you at least call and make sure daddy's okay? please hurry. yaay! hurrts! it hurrrrts!
0.072 yayyy! well i thought it was safe-- i want to look around daddy-- may i? to remember.

(b) *age* (category): child

Figure 4.1: Attention visualization for *profession* and *age* attributes on MovieChAtt.

0.084 i dated her, unfortunately life took us in two different directions and we both ended up married to different people. after two years both of those marriages failed and we found ourselves consoling each other through divorces

0.083 to my ex wife: i apologize for marrying you, i never should have. i never really loved you the way a husband should, you didn't deserve that in all reality i did treat you like a princess though because i am a great person

0.031 she is by no means overweight like you are implying we want to and actually have free access to a hotel pool. she is my fiance. the plan is to see a doc about it

(a) *family status*: married

Figure 4.2: Attention visualization for *family status* attributes on RedDust.

4.6.4 Case study on attention weights

In order to illustrate the types of terms the models are looking for, we display $\text{HAM}_{\text{2attn}}$'s term and utterance weights for *profession* and *age* attributes (on the MovieChAtt dataset) in Figure 4.1, as well as *family status* attribute (on RedDust) in Figure 4.2. While $\text{HAM}_{\text{2attn}}$ is often outperformed by $\text{HAM}_{\text{CNN-attn}}$, this model is more interpretable because individual terms are considered in isolation.

When predicting *military personnel* as the *profession* (Figure 4.1a), the model focuses on military terms such as '*mission*', '*guard*', '*barracks*', and '*colonel*'. When predicting *child* as the *age category* (Figure 4.1b), on the other hand, the model focuses on terms a child is likely to use, such as '*pet*', '*mommy*', and '*daddy*'. According to Reddit posts, *married* users were identified through terms such as '*dated*', *fiance*' and '*divorces*', along with obvious terms like '*marrying*' and '*marriages*' (Figure 4.2a). These examples illustrate how the model is able to infer a speaker's attribute by aggregating signals across utterances.

profession	**significant words**
scientist	characteristics, theory, mathematical, species, changes
politician	governors, senate, secretary, reporters, president
detective	motel, spotted, van, suitcase, parked
military personnel	captured, firepower, guard, soldiers, attack
student	playing, really, emotional, definitely, unbelievable
photographer	xavier, leonard, collins, cockatoo, burke
waiter	rape, stalkers, murdered, overheard, bothering

Table 4.7: Top-5 words from $\text{HAM}_{\text{2attn}}$ characterizing each profession.

In addition to looking at specific utterances, we investigated which terms the model is strongly associating with a specific attribute. To do so, we computed attribute value probabilities for each term in the corpus, and kept the top terms for each attribute value. The results using $\text{HAM}_{\text{2attn}}$ are shown in Table 4.7, which is divided into words that appear informative (top section) and words that do not (bottom section). In the case of informative words, there is a clear relationship between the words and the corresponding profession. Many of the uninformative words appear to be movie-specific, such as names (e.g., '*xavier*', '*leonard*') and terms related to a *waiter's* role in a specific movie (e.g., '*rape*', '*stalkers*'). Reducing the impact of setting-specific signals like this is one direction for future work.

Models	profession		gender		age	
	MRR micro / macro	AUROC	Acc	AUROC	MRR micro / macro	AUROC
HAM $_{\text{CNN-attn}}$	0.19 / 0.18	0.58	0.56	0.58	**0.57** / **0.54**	**0.69**
HAM $_{\text{2attn}}$	**0.21** / **0.21**	**0.67**	**0.61**	**0.64**	0.45 / 0.41	0.45

Table 4.8: Transfer learning performance of pre-trained RedDust models on MovieChAtt.

Models	profession		gender		family status	
	MRR micro / macro	AUROC	Acc	AUROC	Acc	AUROC
HAM $_{\text{CNN-attn}}$	0.20 / 0.16	0.58	**0.52**	0.50	**0.74**	**0.74**
HAM $_{\text{2attn}}$	**0.21** / **0.18**	**0.71**	0.51	**0.54**	0.62	0.64

Table 4.9: Transfer learning performance of pre-trained RedDust models on PersonaChat.

4.6.5 Insights on transfer learning

To investigate the robustness of our trained HAMs, we tested the best performing models (i.e., HAM $_{\text{2attn}}$ and HAM $_{\text{CNN-attn}}$) on a transfer learning task between our datasets. Specifically, we leveraged user-generated social media text (RedDust posts) available in abundance to train the models and subsequently performing inference on the speech-based dialogues (MovieChAtt and PersonaChat). We report the results in Tables 4.8 and 4.9 respectively.

While the scores on PersonaChat are low compared to those in Tables 4.2, 4.1 and 4.3, the HAMs' performance on MovieChAtt is often comparable with the baselines' performance in Tables 4.2, 4.1, 4.4 and 4.3. This difference may be caused by the fact that PersonaChat is a smaller, more synthetic dataset, as discussed in Section 4.6.1.

On MovieChAtt with the *profession* attribute, both HAMs match the performance of all six baselines in terms of macro MRR. Similarly, HAM $_{\text{2attn}}$ matches the performance of five of the six baselines on the *gender* attribute (accuracy), and HAM $_{\text{CNN-attn}}$ matches the performance of four of the six baselines on the *age* attribute (macro MRR). The methods do not perform as well in terms of micro MRR, which may be due to the substantially different attribute value distributions between datasets. Particularly for the *profession* attribute, the lower performance can be explained by missing training instances for certain professions in the RedDust dataset, such as *astronaut* or *monarch*. Improving HAMs' transfer learning performance is a direction for future work.

4.6.6 Profession misclassification study

In this section we investigate common misclassifications on the MovieChAtt dataset for the *profession* attribute, which is the most challenging attribute with the most possible values. A confusion matrix for HAM $_{\text{2attn}}$ is shown in Figure 4.3. Dotted lines indicate several interesting misclassifications: *policemen* are often confused with *detectives* and *special agents* (red line); *scientists* are confused with *astronauts* (yellow line), because sci-fi films often feature characters who arguably serve in both roles; and a *child* is often labeled as a *student*

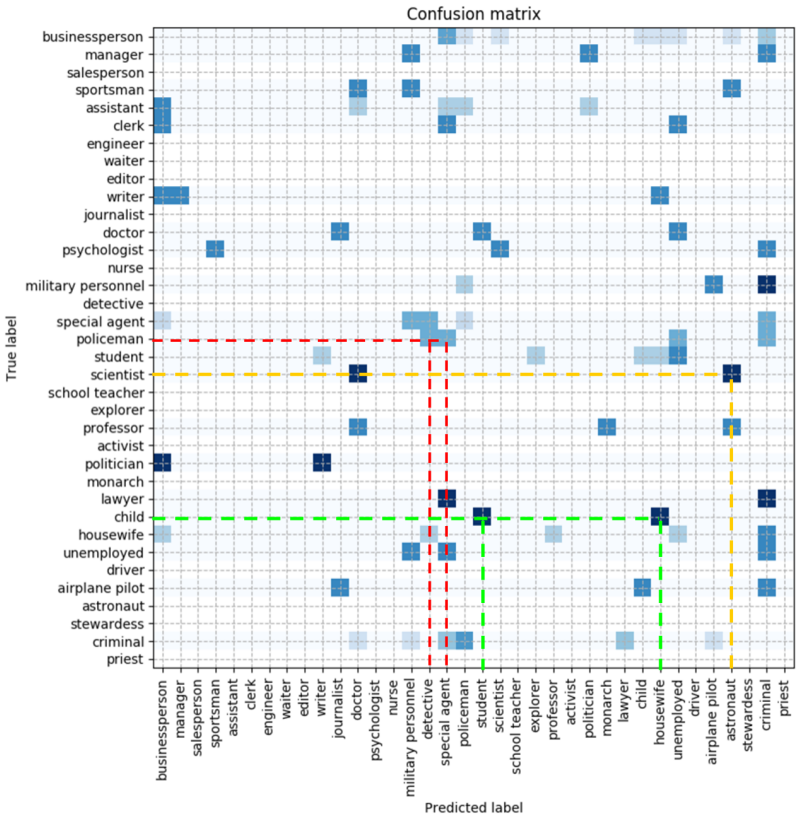


Figure 4.3: Confusion matrix computed with $\text{HAM}_{\text{2attn}}$. True positives are not shown. Darker squares indicate more misclassifications.

or a *housewife* (green line) because they sometimes use similar terms (e.g., '*school*' is used by both children and students, and '*mommy*' is used by both children and housewives). Finally, many occupations are confused with *criminal*, which is the most common profession in MovieChAtt.

4.7 Conclusion

In this chapter we proposed Hidden Attribute Models (HAMs) for inferring personal attributes from conversations. We demonstrated the viability of our approach in extensive experiments considering several attributes on three datasets with diverse characteristics: Reddit discussions, movie script dialogues and crowdsourced conversations. Furthermore, we used an oracle approach to demonstrate that pattern matching is insufficient for extracting personal attributes from conversations, because such attributes are rarely explicitly mentioned. We compared HAMs against a variety of state-of-the-art baselines, showing that HAMs achieve substantial improvements over all of them. We also demonstrated that the attention weights assigned by our methods provide informative explanations of the computed output labels.

As a stress test for our methods, we investigated transfer learning by training HAMs

on one dataset and applying the learned models to other datasets. Although we observed degradation in output quality, compared to training on in-domain data, it is noteworthy that the transferred HAMs matched the performance of the baselines when trained on in-domain data.

4.7.1 Limitations and future work

In this section we enumerate possible directions for future work, some of which are motivated by the limitations of the approach proposed in this chapter.

- **Limited attribute value lists.** Many personal attributes, such as *hobby* or *profession*, have long lists of possible values. In our experiments on predicting professions with HAMs we carefully selected the most popular occupations, which have sufficient amount of labeled examples. This approach misses out on many rare attribute values, for which the training examples are scarce or even not available. Moreover, a significant amount of human supervision is required to manually refine the attribute value lists. In the next chapter we propose our solution to this issue.

- **Improved transfer learning.** Our experiments with HAMs were based on the assumption that the input data resembles actual conversations. However, this is far from being true: the dialogues in MovieChAtt are based on fictional events and sound metaphorical, PersonaChat conversations are artificial and very short, discussion threads in RedDust are distinct from face-to-face offline dialogues. Therefore, there is a genuine need to devise new models having strong transfer learning abilities, so that the method trained on the richly annotated artificial datasets would be able to make proper inference on real-life conversations.

- **Combined prediction of several attribute values.** Most personal attributes are interdependent (e.g., a 6-year-old person cannot be a manager or be married). Thus, the method for predicting personal attributes should ultimately leverage the relationships between multiple attributes by extracting them with a single model. This facilitates the extension of the architecture to any new personal attribute of interest and saves computational resources for training separate models for each attribute.

Chapter 5

Conversational Hidden Attribute Retrieval Model

Contents

Personal knowledge about users' professions, hobbies, favorite food, and travel preferences, among others, is a valuable asset for individualized AI, such as recommenders or chatbots. Conversations in social media are a rich source of data for inferring personal facts. Prior work developed supervised methods to extract this knowledge, but these approaches can not generalize beyond attribute values with ample labeled training samples. Such models are thus inapplicable for the *long-tailed* personal attributes, such as *hobby*, when there is little chance of acquiring labeled training data for the rare attribute values. We overcome this limitation by devising CHARM: a zero-shot learning method that creatively leverages keyword extraction and document retrieval in order to predict attribute values that were never seen during training. Experiments with large datasets from Reddit show the viability of CHARM for open-ended attributes, such as *profession* and *hobby*.

5.1 Introduction

Motivation. Personal knowledge bases capture user traits for customizing downstream applications like chatbots or recommender systems [8]. To provide recommendations that are tailored to the fine-grained characteristics and interests of the user, a PKB should contain personal attributes with a wide range of possible values, such as hobbies, professions, cities visited, medical conditions (experienced by the user) and many more.

A potentially automatic way to populate a PKB with such attributes is to draw them from the user's conversations in social media and dialogues on other platforms. However, a large number of personal attributes and their respective values makes this a challenging task. In particular, there is little hope to have training data for each of these key-value pairs. Moreover, the textual cues in user conversations are often implicit and thus difficult to learn.

Example. Consider the user's utterance: *"I just visited London, which was a disaster. My hotel was a headache and I spent half the time in bed with a fever... So glad to be back home finishing the masts on my galleon."*

As humans, we can infer the following attribute-value pairs: (a) *cities visited:London*, (b) *symptom:fever*, (c) *hobby:model ships*. However, with both implicit and explicit signals present, capturing such user traits is a daunting task. We need to consider the context *"spent in bed with"*, to infer that *fever* relates to a disease (as opposed to *headache*). To predict the user's hobby *model ships*, we have to pay attention to the cues '*galleon*' and '*mast*'. Proper inference requires both deep language understanding and background knowledge (e.g., about ships, cities, etc.).

State of the Art and its Limitations. Explicit mentions of attribute-value pairs can be captured by pattern-based methods [85, 176]. Such methods are able to extract '*London*' from the the previous example by using the pattern *"I . . . visited ⟨city_name⟩"*. Pattern-based approaches are limited, though, by their inability to consider implicit contexts, such as *"finishing the **masts** on my **galleon**"*. Question answering methods can be used to relax rigid patterns [79], but still rely on explicit mentions of attribute values.

In this work we aim to extract attribute values leveraging both explicit and implicit cues, such as inferring *symptom:fever* and *hobby:model ships*. Additionally, we address the cases where there is a long-tailed set of values for such attributes as *hobby*. In principle, deep learning is suitable for such inference [123, 130, 152], but it critically hinges on the availability of labeled training samples for every attribute value that the model should predict.

Supervised training is suitable for a pre-specified limited-scope setting, such as learning personal interest from a fixed list of ten movie genres, but it does not work for the situation with large and open-ended sets of possible values, for which there is little hope of obtaining comprehensive training samples. Therefore, we pursue a *zero-shot learning* [74, 113] approach that learns from labeled samples for a small subset of labels (i.e., attribute values in our setting) and generalizes to the full set of labels including values *unseen* at training time.

Problem Statement. For a given attribute we consider the set of *known* values V, which can be drawn from lists in dictionary-like sources, such as Wikipedia. At training time, our method requires samples for a small subset of values $S \subset V$. Typically, the complement

$V \setminus S$ is much larger than S: $|V \setminus S| \gg |S|$. For instance, S may consist solely of the popular values *sports, travel, reading, music, games*, whereas the complement includes hundreds of long-tail values, such as *beach volleyball*, *model ships*, *brewing*, etc. At inference time we need to predict values from all of V, although most of the values are *unseen* during training.

Approach and Contributions. We present a **C**onversational **H**idden **A**ttribute **R**etrieval **M**odel (CHARM) for inferring attribute values in a zero-shot setting [154]. CHARM identifies cues related to a target attribute, which it then uses to retrieve relevant texts from external document collections, indicative of different attribute values. These external documents could be gathered by simple web search. They help CHARM to link the cues in the user's utterances to the actual attribute values to predict.

CHARM consists of two components: *(i)* a *cue detector*, which identifies attribute-relevant keywords in a user's utterances (e.g., '*galleon*'), and *(ii)* a *value ranker*, which matches these keywords against documents that indicate possible values of the attribute (e.g., *model ships*). Attribute values predicted by CHARM must be *known* but CHARM does not require all values to be *seen* during training.

To evaluate our approach, we conduct experiments predicting Reddit users' professions and hobbies based on their conversational utterances. We demonstrate that CHARM performs well when inferring unseen values and performs competitively with the best-performing baselines when predicting values seen during training. CHARM can easily be extended to other attributes with long-tail values, such as *favorite cuisine*, *preferred news topics* or *medication taken*, by providing a list of known attribute values, training examples for a subset of these values and access to external documents (e.g., via a Web search engine).

The salient contributions of this work are:

- a method for inferring both seen and previously unseen (zero-shot) attribute values from a user's conversational utterances;
- a comprehensive evaluation for the *profession* and *hobby* attributes over a large dataset of Reddit discussions; and
- a demonstration platform[1] showcasing the ability of CHARM to make predictions from dialogue with the user or the user's social media submissions [155].

5.2 Related work

Zero-shot learning. CHARM is designed for handling attribute values that were never seen at training time – a *zero-shot learning* problem, extensively studied in the field of computer vision but less explored in NLP. A technique employed in CHARM is similar to the approach proposed by Ba et al. [5] for visual classes, which builds image classifiers directly from encyclopedia articles without training images.

Most zero-shot studies for NLP [163] deal with machine translation, cross-lingual retrieval and entity/relation extraction. For example, in *relation extraction* task [79] the relations serve as unseen classes and their instances are recovered by casting the relations to natural language

[1] https://d5demos.mpi-inf.mpg.de/charm

questions and reducing the problem to reading comprehension; in *entity extraction* [114] the extraction of entities from the web documents is done with a text query, removing the need for specifying a set of seed terms. Methods proposed in [79, 114] are not suitable for our task, because they identify values that are explicitly mentioned, rather than inferring them. Our task is similar to zero-shot *text classification* [175, 181], where the class labels are represented as single-word embeddings.

Other zero-shot models solving natural language problems include the ones for text filtering and classification. Li et al. [80] solve the task of zero-shot *document filtering* by learning relevance between categories and documents, which are represented with category-dependent embeddings. Dauphin et al. [30] investigates zero-shot *semantic utterance classification*, by mapping the utterances and potentially unseen categories into the same semantic space, where they can be matched with distance functions.

Keyword extraction from conversational text. Keyword or keyphrase extraction concerns the task of automatic selection of important terms from a document to represent its content. The extracted terms are beneficial for many applications including document indexing, summarization and classification.

Owing to its importance, several extraction methods have been extensively studied and evaluated on various corpora, mostly on news articles, web documents and scientific/technical reports. Less attention has been given to keyword extraction from conversational texts, such as meeting transcripts [53, 90], live chats [69] and social media posts [171, 182, 185]. Notable applications of keyword extraction from conversations include generating personalized tags for Twitter users [171], searching for relevant email attachments [52] and just-in-time information retrieval [53].

Prior work mostly pursued unsupervised keyword extraction approaches [106, 136], due to limited availability of training data. Few studies use supervised learning, with feature-based classifiers [69] or neural sequence tagging models [182]. Our neural approach for keyword detection lies in between, as we learn to identify salient keywords for a specific attribute (e.g., *profession*), without having training data of relevant keywords.

Unsupervised techniques for keyword extraction can generally be split into several categories [54], notably: *(i) statistical*, which are based on simple word features, such as term frequency or relational position in the document [15, 129], word co-occurrence [102] or keyphrase co-occurrence counts [136]; and *(ii) graph-based*, which utilize a word/phrase graph constructed from the document and extract keywords using graph ranking methods [13, 106]. We consider two unsupervised keyword extraction architectures as baselines: statistical method *RAKE* [136] and graph-based model *TextRank* [106].

Information Retrieval in NLP. Most existing work leveraging information retrieval components to solve NLP tasks focused on question answering [50, 73, 161] or dialogue systems [37, 95], where the retrieval part is responsible for ranking the most appropriate answers or responses, given a question or chat session. As far as we know, we are the first to leverage a retrieval-based model for inferring attribute values without training samples.

Reinforcement Learning in NLP. Reinforcement learning (RL) methods are often applied in conversational models for response generation [63], based on the feedback from human quality assessment scores for the output utterances. Other NLP tasks where reinforcement

learning can be applied include question answering [92, 127], text classification [184] and entity linking [35].

Several studies use RL decision processes to pick up relevant words or phrases from the input texts, which is close to our work. For example, Wang et al. [162] proposed to perform aspect-level sentiment classification using reinforcement learning to select segments of texts on which the sentiment should be predicted. Chen et al. [19] detects events and their corresponding keywords from Twitter texts using an RL method to select posts, which talk about events and extract keywords from them. Zhang et al. [184] proposed a reinforcement learning agent that selects the words which should be removed from the sentences, creating a concise sequence representation.

5.3 Background

Training CHARM involves applying a non-differentiable `argmax` operation, which prevents using end-to-end backpropagation. In this section we provide the technical background on the *policy gradient method*, which we use to mitigate this problem.

Reinforcement learning is a machine learning technique based on training an intelligent agent to maximize the reward by selecting appropriate actions. The agent interacts with the *environment* by observing its current state and taking actions; as a result the agent gets the feedback from the environment, which serves as a signal about the correctness of the chosen action. Reinforcement learning environment can be formulated as a *Markov Decision Process*, a tuple (S, A, P, r, γ), where

- S is a finite set of states,
- A is a finite set of actions,
- $T(s, a) = s' : S \times A \to S$ is the *transition function*, specifying the mapping from the current state s and the taken action a to a new state s',
- $r(s, a) : S \times A \to \mathbb{R}$ is the *reward function*, specifying the real-valued reward for taking the action a while being in state s,
- $\gamma \in (0, 1]$ is the discounting factor, representing the decrease of the reward for the actions further in the future.

A stochastic *policy* $\pi(a_t|s_t)$ is a function, specifying the probability of taking action a_t while being in the state s_t at the timestep t. A *trajectory* $\tau = [s_t, a_t]_{t=0}^{T}$ is a sequence of (state, action) steps, induced by the policy π, where T denotes the terminal step. The *expected reward* J is defined as:

$$J(\theta) = \mathbb{E}_{\tau \sim p_\pi(\tau)} \sum_{t=0}^{T} \gamma^t r(s_t, a_t|\pi) \tag{5.1}$$

where the expectation is computed with respect to possible random trajectories $\tau \sim p_\pi(\tau)$, determined by the policy function. The expected reward defines the mathematical expectation of the total gain until timestep T by following the policy π.

Policy gradient methods solve reinforcement learning problems, which have parameterized differentiable policy function π_θ, by maximizing the expected reward $J(\theta)$ with respect to θ. This is done by differentiating $J(\theta)$ using the following formula:

$$\nabla_\theta J(\theta) = \sum_{t=1}^{K} R_t \nabla_\theta log \pi_\theta(a_t|s_t) \tag{5.2}$$

where R_t is the discounted future cumulative reward:

$$R_t = \sum_{t'=t+1}^{T} \gamma^{t'-t-1} r_{t'} \tag{5.3}$$

The policy π could be implemented by a neural network, which parameters θ are to be updated using policy gradient methods. For example, this can be done with the REINFORCE algorithm [168]. In REINFORCE the reinforcement learning agent generates a random trajectory under the current policy π_θ and then, for every timestep in the trajectory $t = 1, ..., T$ the policy parameters are updated as follows:

$$\theta_{new} = \theta_{old} + \alpha \nabla_\theta R_t log \pi_\theta(a_t|s_t) \tag{5.4}$$

5.4 Methodology

In this section we describe CHARM, our proposed model for inferring personal attributes from conversations. As illustrated in Figure 6.2, CHARM's operation consists of two stages: *cue detection* and *value ranking*. As input CHARM receives the user's utterances $U = u_0, ..., u_N$ that contain a set of terms $t_0, ..., t_M$, for example, U ={*"I stayed late at the* ***library*** *yesterday", "****Studied*** *for the* ***exam*** *so I could have better* ***grades*** *than my* ***classmates***"}. In the first stage, the term scoring model assigns a score to each term in the user's utterances, yielding $l_0, ..., l_M$. The highest scoring terms are then selected to form a query $Q = q_0, ..., q_K$, characterizing the user's correct attribute value, e.g., Q ="*library studied exam grades classmates*" for the *profession* attribute.

In the second stage, Q is evaluated against an external document collection $D = d_0, ..., d_L$; each document in D is associated with possible attribute values. Documents such as *Wiki:Student* and *Wiki:Dean's List*[2], which are associated with the attribute value *student*, would score high with the example query. The score aggregator then ranks the attribute values based on the documents' scores $s_0, ..., s_L$, for instance, yielding a high attribute score for *student* given our example utterances. The list of attribute values V is **known** in advance (e.g., taken from Wikipedia list of professions); however, potentially only a subset of values $S \subset V$ have instances **seen** during training.

The rank of the correct attribute value acts as a distant supervision signal, allowing us to train term selection, regardless of the non-differentiable `argmax` operation. CHARM is trained using reinforcement learning via the REINFORCE policy gradient method described in the previous section.

[2]Wikipedia pages: Student & Dean's_list

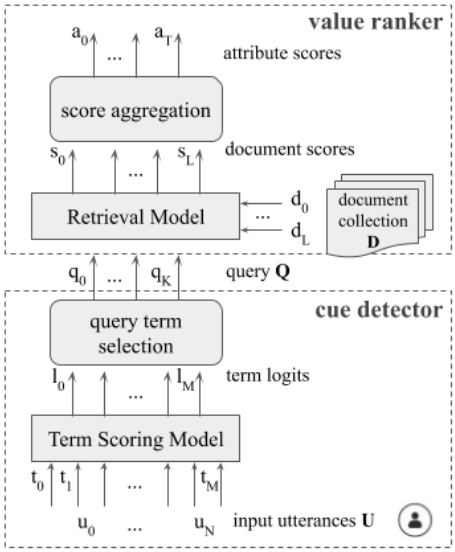


Figure 5.1: The pipeline of CHARM. The Term Scoring Model assigns scores $l_0 ..., l_M$ to the terms in the input utterances $u_0, ..., u_N$. The terms with the highest scores are passed to the Retrieval Model, which queries the document collection D. The document scores are aggregated to produce attribute value scores for predictions.

5.4.1 Cue detection

The term scoring model δ evaluates how useful each word in a given user's utterances is for making a prediction, and assigns real-value scores $l_0, ..., l_M$ to the terms accordingly. That is, $l_j = \delta(t_j | t_0, ..., t_M; W)$, where W denotes the parameters of the model. The term scores $l_0, ..., l_M$ are then used to select the words which will form the query for the value ranking component.

Due to our use of REINFORCE, the selection process differs between the training and prediction settings. During training, the scores $l_0, ..., l_M$ are normalized with a softmax function to obtain probabilities $p_0, ..., p_M$, which are used to incrementally sample without replacement a query consisting of K terms. The query length K is a hyperparameter, optimized in the grid search. The sampling in effect allows the words with low scores a better chance to be selected, thus encouraging exploration. At inference time the query is formed by taking the terms with K top scores.

The term scoring model should produce high scores for terms that are descriptive of the user and of the attribute in general, instead of a specific attribute value. This means that it should be able to exploit background knowledge and a term's context to judge its relevance to the attribute. For instance, having seen the phrase *"stayed late at the hospital"* for the *physician* at training time, at prediction time an ideal model would correctly estimate the importance of the word '*library*' in the phrase *"stayed late at the library"*, even if there were no instances of *student* in the training set. Considering this, we give preference to the models that operate on sequences, as opposed to the bag-of-word models. We select BERT [32] as our term scoring model, because it is a sequential model incorporating world knowledge that should effectively use word context, predicting whether terms are related to an attribute.

For further description, let us suppose the cue detector picks the words $Q = q_0, ..., q_K$ as the query terms for CHARM's value ranking stage. A typical query would consist of the terms associated with the correct attribute value (for example, $Q =$"*library studied exam grades classmates*").

5.4.2 Value ranking

The second stage of the model consists of two steps: first, using the selected query terms to rank the documents in the external collection; and second, aggregating document scores to predict values.

Document ranking. The ranking component takes two inputs: query terms $Q = q_0, ..., q_K$ resulting from the cue detector and an (automatically labeled) document collection $D = d_0, ..., d_L$. The document collection could be a set of Web pages, where each page indicates a specific attribute value, $v_0, ..., v_L$. For example, by generating a search-engine query "*hobby* ⟨*value*⟩" we can gather web pages related to specific hobbies.

The ranker $\rho(Q, d_k)$ evaluates the query Q, constructed by the cue detector, against each document d_k in the document collection to produce document relevance scores $s_0, ..., s_L$. For the example query "*library studied exam grades classmates*s", the document *Wiki:Dean's List* labeled with *student* will get a higher score than *Wiki:Junior doctor* (for *physician*).

We consider two particular instantiations of the ranker: BM25 [135] and KNRM [173], described in Chapter 2.2.6. BM25 is a strong unsupervised retrieval model, whereas KNRM is an efficient neural retrieval model that can consider semantic similarity via term embeddings in addition to considering exact matches of query terms.

Document score aggregation. The document scores $s_0, ..., s_L$ obtained from the ranker are then aggregated to produce scores for each known attribute value. Depending on the document collection used, each attribute value may be represented by several documents. For example, the *student* attribute value may be associated with documents *Wiki:Dean's List*, *Wiki:Master's degree*, etc. In this case, the scores per document have to be aggregated to form the final scores $a_0, ..., a_T$ for each attribute value in V. In our experiments, we consider the following aggregation techniques: *(i) average* (which allows multiple documents to contribute to the final ranking) and *(ii) max* (which may help when the document collection is noisy and we care only about the top-scoring document for each value). Having obtained the final attribute scores $a_0, ..., a_T$, we sort them to get the top value as the model's prediction.

5.4.3 Training

While predicting attribute values is not inherently a reinforcement learning problem, we utilize the REINFORCE policy gradient method to train the cue detector component because there are no labels indicating which input terms should be selected. This allows the cue detector to be trained based on the correct attribute values regardless of the non-differentiable `argmax` operation needed to identify the K top scoring terms from the scores it outputs.

When using the policy gradient method, the *state* in our system is represented by a sequence of input terms $t_0, ..., t_M$. Each of the M input terms also represents an independent *action*. The term scoring model acts as the *policy*, which outputs the term selection probabilities based on the current state. Then a term is sampled (at training time) or the term with maximum probability is selected (at prediction time) and added to the query.

During training, we form the query by sampling without replacement one word at a time. After sampling each term, we issue the current query and get intermediate feedback. The training episode ends when the query reaches its maximum length K. We define the reward r_i for an intermediate query to be the normalized discounted cumulative gain (the nDCG ranking metric) of the correct attribute values' scores after aggregation at timestep i. The objective of REINFORCE is to maximize $J = \sum_{i=1}^{K} r_i * \log p_i$ by updating the weights of the policy network (where p_i is the probability of selecting a term at timestep i).

5.5 Dataset

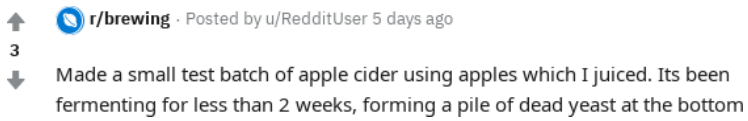


Figure 5.2: Example of an input utterance with cues hinting that *brewing* is the user's hobby.

The datasets used in our experiments cover two types of input: *(i)* users' utterances along with their corresponding attribute-value pairs (e.g., *hobby:brewing* from the example in Figure 5.2), and *(ii)* a collection of documents associated with each attribute value (e.g., documents describing *brewing* as a hobby). We consider two exemplary attributes: *profession* and *hobby*. We define lists of their attribute values based on Wikipedia lists[3].

5.5.1 Users' utterances

As users' utterances we used the dataset of Reddit submissions labeled with weak supervision, described in Section 3.3.5. For our experiments, we removed all posts containing explicit personal assertions that we used for labeling each user, because we want to test the ability of CHARM to predict attribute values based on inference, as opposed to explicit pattern extraction.

For practical reasons, for each attribute we sorted the labeled users by the Snorkel probabilistic labeling model's scores and cropped the set to maximum 500 users per attribute value and 6000 users in total. The final dataset has 23 users per attribute value on average; there are 605 and 245 users who have multiple attribute values for *profession* and *hobby* respectively. Cropping the number of users effectively resulted in reducing the number of attribute values in the original Wikipedia lists, which were fixed as 149 for *hobby* and 71 for *profession*.

[3]Wikipedia pages: List_of_hobbies & Lists_of_occupations

5.5.2 Document collection

The scope of possible attribute values may be open-ended in nature, and thus, calls for an automatic method for collecting Web documents. In this work, we consider three different Web document collections; summary statistics on the number of documents per attribute value are provided in Table 5.1. Each document may be associated with multiple attribute values, which matches the same aspect of the users' labeling. To provide more diversity and comprehensiveness we augmented our predefined lists of known attribute values with their synonyms and hyponyms.

Note that the approaches used to construct the document collections are straightforward and easily applicable for further attributes, such as *favorite travel destination* or *favorite book genre*.

		min	max	avg	total
profession	Wiki-page	1	10	2	156
	Wiki-category	1	191	57	4,156
	Web search	71	100	92	6,688
hobby	Wiki-page	1	1	1	149
	Wiki-category	2	479	74	10,782
	Web search	54	100	82	12,312

Table 5.1: Document collection statistics.

Wikipedia pages (Wiki-page). To create this collection we took the lists of known attribute values and automatically retrieved a Wikipedia page corresponding to each value, which usually coincides with the article title (e.g., *Wiki:Barista*).

Wikipedia pages–extended (Wiki-category). This collection is an extension of Wiki-page that additionally includes pages found using Wikipedia categories. This allows us to include pages about concepts related to the attribute values, such as tools used for a profession and the profession's specializations. To construct Wiki-category, we identified at least one relevant category for each attribute value and included all leaf pages under the category (i.e., including no subcategories). For example, we label all pages under *Wiki category:acting* with profession value *actor*.

Web search. To create this collection we queried a Web search engine using attribute-specific patterns: `my profession as <profession value>` and `my favorite hobby is <hobby value>`. The collection consists of the top 100 documents returned for each value. Such patterns can be created with low effort by evaluating a few sample queries. Alternatively, patterns could be mined from a corpus or simplified to the generic form `<attribute> <value>`.

5.6 Experimental Setup

We evaluate the proposed method's performance in two experimental settings. First, we consider a zero-shot setting, in which the attribute values in the training and test data are completely disjoint (i.e., the test set only contains *unseen* labels). This setting evaluates

how well CHARM can predict attribute values that were not observed during training. Second, we consider the standard classification scenario, in which all attribute values are *seen* as labels in both training and test sets. This demonstrates that CHARM's performance in a normal classification setting does not substantially degrade because of its proposed architecture.

Experimental setup details differ for these two evaluation settings, which will be discussed in the following subsections. All our models were implemented in PyTorch; the code and data are available at https://github.com/Anna146/CHARM.

Training and test data. For the *unseen* experiments, we perform ten fold cross-validation with folds constructed such that each attribute value appears in only one test fold. Each of the folds contains roughly the same number of users and approximately 2-4 unique attribute values[4]. We assigned the users having multiple attribute values to a fold corresponding to one of their randomly chosen values. For the experiments with *seen* values, we randomly split the users into training and test sets in a 9:1 proportion, respectively, which yielded 5232/580 users for *profession* and 5246/582 for *hobby*.

Hyperparameters. BERT, the term selection component, generates a contextualized embedding for each input term, which we process with a fully connected layer to produce a term score for each word in its context. Specifically, we use the pre-trained BERT model (bert-base-uncased) model with 12 transformer layers. To reduce BERT's computational requirements, we discard the last 6 transformer layers (i.e., we use embeddings produced by the earliest 6 layers) after observing in pilot experiments that this outperformed a distilled BERT model [137].

Following prior work [57], KNRM was trained with frozen word2vec embeddings on data from the 2011-2014 TREC Web Track with the 2009-2010 years for validation. We initialize KNRM with these pre-trained weights.

During training, we sample 5 negative labels (i.e., incorrect attribute values) to be ranked when calculating the nDCG reward. For each label, we sample a subset of 15 documents to represent the label (i.e., attribute value). If the document collection has fewer than 15 documents for a label (e.g., Wiki-page), we consider all the label's available documents. When making predictions, we consider all documents and all labels. In both settings, we truncate documents to 800 terms when using KNRM for efficiency and use the full documents with BM25. We optimize the following hyperparameters in a grid search: (*i*) document aggregation strategy (*average* vs *max*); (*ii*) length of query; and (*iii*) maximum number of epochs. The best hyperparameters were chosen based on the MRR score.

Baselines. For the *unseen* experiments, we evaluate CHARM's performance against an end-to-end BERT ranking method and against a BM25 [134] ranker combined with two state-of-the-art unsupervised keyword extraction methods: TextRank [106] and RAKE [136]. We additionally include a baseline giving the user's full utterances as input to BM25 (baseline: *No-keyword*).

Following related work [27, 111], we train the BERT IR baseline using the binary cross-entropy loss to predict the relevance of each document to the user's utterances (acting as

[4]We used a greedy algorithm to approximate a solution to the NP-hard bin packing problem.

	train (10.000 instances)	**test** (100 instances)
CHARM $_{\text{KNRM}}$	31.8	1.2
CHARM $_{\text{BM25}}$	54.4	10.9
BERT IR	56.2	72.7

Table 5.2: Running time of the models given in minutes. The train time is a sum of the times across all training epochs, all times are averaged across folds in the unseen experiment.

queries). We use the same pre-trained BERT model as in CHARM. To fit both utterances and documents into the input size of BERT, we split both into 256-token chunks and run BERT on their Cartesian product. To obtain the final score for each utterances-document pair we average across all chunk pairs. Given N utterances and M documents, this baseline processes $N \times M$ inputs with BERT, whereas CHARM processes N inputs with BERT and M inputs with an efficient ranking method. This makes the BERT IR baseline very computationally expensive on the Wiki-category and Web search document collections, which contain 4,000-12,000 documents. In order to run the baseline on these collections, we sample three documents per label; even with this change, BERT IR is 60x slower than CHARM. More details on the models' running time are in Table 5.2.

For the *seen* experimental setup, we compare CHARM with state-of-the-art supervised approaches for inferring attribute values:

- *New Groningen Author-profiling Model (N-GrAM)* [9] exploits a linear Support Vector Machine (SVM) classifier [26] that utilizes character n-grams ($n = 3, 4, 5$) and term n-grams ($n = 1, 2$) with sublinear TF-IDF weighting as features.

- *Neural Clusters (W2V-C)* [123] were obtained by applying spectral clustering ($n = 200$) on a word similarity matrix, computed via cosine similarity of pre-trained word embeddings. The ratio of words from each cluster is then used as feature vectors for a Gaussian Process (GP) classifier [22], which is the best reported classification model for the task [123].

- *Convolutional Neural Network (CNN)* [10] was proposed for the task of predicting the age and gender of Twitter users. CNN was applied to individual utterances, and the majority classification label is used as the prediction per user.

- *Hidden Attribute Models*, described in Chapter 4. For the experiments discussed in this chapter we used a HAM $_{\text{2attn}}$ variant of HAMs, as it outperforms the other variants in the majority of cases.

Additionally we consider a fine-tuned supervised BERT model that performs attribute value classification using its `[CLS]` representation. In the *seen* experimental setup the baseline models are single-value, therefore, we split every multi-value user into several inputs through all their attribute values.

Evaluation metrics. Following prior work on personal attribute inference [123, 152], we

Model	*profession*						*hobby*					
	Wiki-page		Wiki-category		Web search		Wiki-page		Wiki-category		Web search	
	MRR	nDCG	MRR	nDCG	MRR	nDCG	MRR	nDCG	MRR	nDCG	MRR	nDCG
No-keyword + BM25	.15*	.32*	.17*	.37*	.11*	.28*	.16*	.42*	.13*	.35*	.06*	.22*
RAKE + BM25	.16*	.33*	.19*	.39*	.11*	.28*	.17*	.42*	.14*	.37*	.07*	.23*
RAKE + KNRM	.16*	.33*	.13*	.34*	.15*	.34*	.12*	.32*	.12*	.31*	.06*	.24*
TextRank + BM25	.21*	.39*	.26*	.45*	.15*	.32*	.21	.46	.20*	.42*	.10*	.28*
TextRank + KNRM	.21*	.38*	.18*	.36*	.20*	.40*	.15*	.36*	.16*	.36*	.11*	.31*
BERT IR	**.30**	.45	.28*	.44*	.26*	.38*	.22	.43*	.18*	.42*	.15*	.33*
CHARM$_{\text{BM25}}$	.29	**.46**	.28*	.47*	.28*	.45*	**.24**	**.47**	.21*	.43*	.11*	.30*
CHARM$_{\text{KNRM}}$	.27	.44	**.35**	**.55**	**.41**	**.59**	.22	.44*	**.27**	**.49**	**.19**	**.38**

Table 5.3: Results for *unseen* values. Results marked with * significantly differ from the best method (in bold) measured by a paired t-test ($p < 0.05$). As described in the experimental setup, BERT IR on Wiki-category and Web search must consider a subset of documents.

consider ranking metrics MRR and nDCG, as they are the most informative for predicting the labels of the attributes with many possible values. Given that MRR assumes there is only one correct attribute value for each user, we calculate MRR independently for each attribute value before averaging; nDCG is averaged over users.

5.7 Results

5.7.1 Quantitative Results

***Unseen* values (zero-shot mode).** The models' performance evaluated only on the attribute values that were not observed during training is shown in Table 5.3. Both CHARM variants significantly outperform all unsupervised keyword-extraction baselines for both attributes on all document collections. This suggests the importance of training the cue detector to identify terms related to the attribute, instead of the more general keywords usually given by unsupervised keyword extractors. BERT IR performs similarly to CHARM for the Wiki-page dataset, but shows significantly worse results for the remaining datasets, taking approximately 60x longer than CHARM$_{\text{KNRM}}$ to perform inference.

Interestingly, the *No-keyword* method performs on par with the other baselines. It shows that the words produced by the state-of-the-art keyword extraction models are not more helpful than the ones automatically selected by TF-IDF scores in BM25 model, highlighting the difficulty of keyword extraction from conversational data.

For both attributes, CHARM$_{\text{KNRM}}$ always outperforms the BM25 variant on Wiki-category and Web search collections. This may be related to the size of document collections, which allows for more variations in the vocabularies that are captured well by the term embeddings in KNRM. Another observation is that for CHARM$_{\text{KNRM}}$, while Web search yields the best result for *profession*, Wiki-category is the best collection for *hobby*, possibly due to the noisy hobby-related documents from web search. CHARM$_{\text{BM25}}$ on Wiki-page does not require any additional inputs and consistently performs as well as or better than the baselines across both attributes. Wiki-category performs significantly better than all

Model	Document collection	*profession*		*hobby*	
		MRR	nDCG	MRR	nDCG
N-GrAM	-	.13*	.43*	.11*	.40*
W2V-C	-	.09*	.39*	.08*	.32*
CNN	-	.20*	.52*	.14*	.43*
HAM$_{2attn}$	-	.32*	.59*	.33	**.55**
BERT	-	**.50**	**.68**	**.35**	**.55**
CHARM$_{BM25}$	Wiki-page	.42*	.57*	.31*	.51*
	Wiki-category	.38*	.56*	.32	.50*
	Web search	.49	.65	.31*	.51
CHARM$_{KNRM}$	Wiki-page	.37*	.54*	.28*	.46*
	Wiki-category	.43*	.62*	.31	.51*
	Web search	.49	.66	.31	.51

Table 5.4: Results for *seen* values. Results marked with * significantly differ from the best method (in bold face) measured by a paired t-test ($p < 0.05$).

baselines for both attributes, making it a reasonable choice when Wikipedia categories are available.

To demonstrate that the collections are resilient to inaccuracies in their automatic construction, we conducted an experiment where some percentage of the documents' attribute values were randomly changed. We found that randomly changing 20% of the documents' labels resulted in approximately a 15% MRR decrease for CHARM$_{KNRM}$ on Web-search and Wiki-category. The performance decrease on these collections was roughly linear. This indicates that the noise in the document collection does not severely damage CHARM's performance.

***Seen* values (supervised mode).** In this experiment we evaluate CHARM's performance in the fully supervised setting (i.e., all labels are seen during training). From the Table 5.4 we observe that CHARM's performance is competitive compared to HAM$_{2attn}$ (i.e., the best-performing attribute value prediction method from prior work) and the state-of-the-art BERT model. The fully supervised BERT model consistently performs best for both attributes, though these increases are not statistically significant over all CHARM configurations. Furthermore, BERT and HAM$_{2attn}$ are trained with full supervision in this experimental setting, whereas CHARM still uses a policy gradient.

Another observation is that in this experiment the Web search collection consistently performs best, suggesting that the collection's shortcomings are mitigated when all labels are observed.

5.7.2 Qualitative Analysis

Analysis of selected terms For each attribute value, we gathered all query terms for the users predicted as having this attribute value, together with the term scores from the cue detector. We then averaged the scores for each term within an attribute value, and selected top 10 terms as the representative ones. Terms were extracted using CHARM$_{KNRM}$

	profession					
	barista (MRR=0.4, #sample=73)		**screenwriter** (MRR=0.65, #sample=52)		**airplane pilot** (MRR=0.64, #sample=14)	
CHARM	coffee	shop	script	story	pilot	flying
	starbucks	guitar	screenplay	film	flight	teacher
	store	student	screenwriting	films	training	fire
	school	customer	scripts	photo	fly	trading
	manager	college	writing	movie	pilots	military
TextRank	people	amp	first	hollywood	people	american
	first	love	people	tomorrow	first	lots
	coffee	things	thanks	time	things	guy
	today	starbucks	amp	second	today	time
	thanks	work	stuff	one	thanks	guys

Table 5.5: CHARM$_{\text{KNRM}}$'s top 10 terms per label for *profession* attribute, compared with TextRank keywords.

on Wiki-category in *unseen* experiments. We performed the same method for the TextRank keywords, because this was the best performing keyword-based baseline in the *unseen* experiments. The comparison of selected terms by CHARM vs TextRank is reported in Table 5.5 and Table 5.6 for selected attribute values of *profession* and *hobby* respectively.

We can observe that regardless of the small sample size for some values like *airplane pilot*, CHARM can still detect meaningful words. For *barista*, CHARM did not even consider the term '*barista*', but rather focused on the words such as '*coffee*' and '*starbucks*'. Choosing terms like '*screenplay*', '*scripts*' and '*screenwriting*' helps the model to distinguish *screenwriter* from the other film-related professions like *director*.

Picking the terms like '*cake*', '*baking*' and '*bread*', helps the model to distinguish between *baking* and *cooking* hobbies more effectively. Note, that even for rare unusual hobbies like *quilting*, CHARM manages to select indicative terms. This essentially shows that the model can easily be used for large long-tailed lists of attribute values.

For the attribute values where CHARM's MRR scores are considerably high (e.g., *profession:screenwriter*, *hobby:baking*), the detected cues are meaningful, diverse and quite distinctive. On the other hand, for attribute values with low MRR scores, some terms are representative, however, they are also easily confused with other attribute values. For instance, some *model aircraft* hobby terms may also refer to the *air sports* hobby.

Finally, as opposed to CHARM, TextRank keywords rarely make sense. This suggests that unsupervised keyword detectors are not capable of producing useful attribute-value-related keywords from users' utterances.

Misclassification Study To conduct error analysis, we plotted a confusion matrix of CHARM$_{\text{KNRM}}$ in the *unseen* experiment for *profession* attribute, which is shown in Figure 5.3.

We observe that medical professions such as *dentist, nurse, pharmacist* and *surgeon* are

	hobby					
	baking (MRR=0.46, #sample=64)		**quilting** (MRR=0.26, #sample=27)		**model aircraft** (MRR=0.11, #sample=2)	
CHARM	cake	bread	sewing	way	cat	dimensions
	food	cream	quilting	game	plane	pilots
	recipe	cooking	quilt	metal	construction	song
	cheese	pasta	fabric	design	planes	steam
	baking	cook	music	playing	energy	music
TextRank	thanks	things	thanks	today	thanks	work
	first	work	first	science	german	elyrion
	amp	food	things	kids	steam	time
	people	time	people	time	tapjoy	purchase
	recipes	second	amp	lots	motorola	air

Table 5.6: CHARM$_{\text{KNRM}}$'s top 10 terms per label for *hobby* attribute, compared with TextRank keywords.

profession		*hobby*	
firefighter (MRR=0.46)	**investor** (MRR=0.52)	**knitting** (MRR=0.68)	**ice hockey** (MRR=0.68)
Firefighter	Index_fund	Yarn_over	Extra_attacker
Firefighter_assist_and_search_team	Venture_capital	Brioche_knitting	Ice_hockey_rules
Calvert_County_Fire-Rescue-EMS	Treasury_management	Combined_knitting	Neutral_zone_trap
Firefighter_arson	Buy_side	Flat_knitting	Playoff_beard
Fire_captain	Sovereign_wealth_fund	Tunisian_crochet	Line_(ice_hockey)

Table 5.7: CHARM$_{\text{KNRM}}$'s top 5 retrieved documents per attribute value.

often confused to *doctor* in general. Professions associated with studying (*academic, teacher* and *student*), beauty (*hairdresser* and *tattoo artist*) and art (*musician* and *poet*) are often confused with each other. *Salesman* and *accountant* are confused to *broker*, because of the common financial terms used.

Analysis of top ranked documents For each attribute value, we collected all documents that were returned for a user with the given value as the ground-truth label. We then averaged the scores for each document and selected the top 5 retrieved documents from Wiki-category, shown in Table 5.7 for several *profession* and *hobby* attribute values.

It is interesting to observe that in spite of the common lexicon for some similar values, the model manages to retrieve documents which are relevant to a particular value, e.g., documents for *investor* are distinct from other financial-related professions, like *broker* or *salesman*. It is also worth mentioning that the retrieved pages for *investor* and *ice hockey* are rather the pages for related lexicon (e.g., '*venture capital*' and '*playoff beard*' respectively), which shows the ability of CHARM to detect indirect cues.

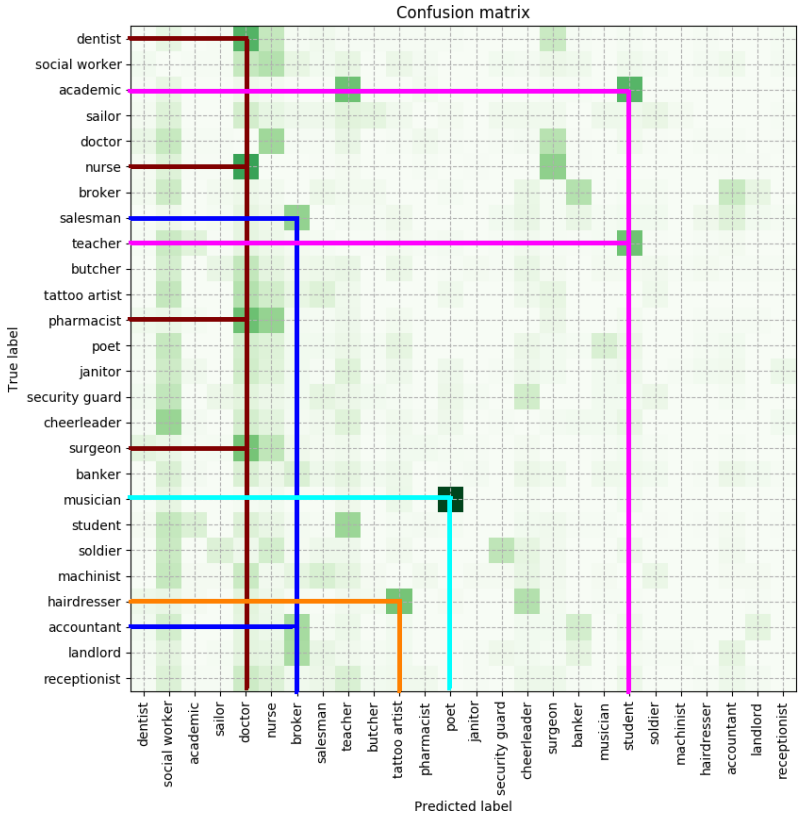


Figure 5.3: Confusion matrix for *profession* with CHARM $_{\text{KNRM}}$ on *unseen* experiments, with some values removed for brevity. Unseen values are aggregated across folds. Darker cells indicate more misclassifications. The lines illustrate misclassifications of interest.

5.8 CHARM Demo

In this section we present a web demonstration platform, accessible at https://d5demos.mpi-inf.mpg.de/charm, that showcases CHARM as a predictive model for extracting personal knowledge from conversational utterances [155]. The contribution of such system is twofold. First, the demonstration can help users protect their privacy by identifying parts of their generated content that could give away personal information. Second, the system shows in detail how the model arrives at the prediction, which is rarely reflected in most automated extraction systems for personal facts.

5.8.1 Motivation

Personal knowledge is a versatile resource that is valuable for a wide range of downstream applications. As observed in Chapter 2, there has been ample research on automatically extracting or inferring personal knowledge. The developed models for conversational data predict a wide range of personal attributes from basic demographics and personality features to fine-grained interests and biography facts.

Such models can benefit many practical applications, yet they potentially endanger privacy. Thus, users should be given an opportunity to assess how the extraction models work in a transparent way. First, this enables users to explore how much personal information can be revealed from what they say online. Second, this helps to explain the reasoning leading to specific personalized ads and recommendations.

To address this issue we develop a demonstration platform for personal knowledge extraction methods, with CHARM as the underlying model. Such setting gives users a chance to directly observe the model's predictions (as opposed to, for example, trying to interpret the recommendations and ads on the websites).

We demonstrate CHARM's predictive capacity in two possible scenarios. The first setting demonstrates how a chatbot can interact with users to collect personal facts, designed as a guessing game. This provides the users with an opportunity to give creative answers and explore the model's capabilities, particularly in inferring the attribute values from given *cues* (e.g., '*pool*', '*paddles*') instead of explicit *mentions* (e.g., '*swimming*'). Users can also try out some rare values (e.g., *quilting*) or test how fine-grained the predictions can be (e.g., *curling* instead of *sports*). The second scenario involves applying CHARM on the real users' posts on social media.

The proposed CHARM demonstration enables the users to *(i)* see what personal information is disclosed by their answers or social media posts, and *(ii)* get explanations on how the prediction was made. This supports users' privacy and model's transparency, which are rarely considered by personalized downstream applications, such as search or recommendation engines.

5.8.2 Demonstration platform

Our demonstration system supports prediction of two personal attributes: *profession* and *hobby*, and incorporates two input scenarios: *chatbot* and *social media* settings.

5.8.2.1 Input scenarios

Chatbot setting. Personal assistants enhanced with background knowledge about their users can give better responses and initiate more interesting conversations. In this setting, we imitate how an intelligent assistant can infer personal facts from interactions with its user without asking explicit questions, such as *"What is your job?"*. The interaction is designed as a game, where the chatbot asks several attribute-related questions, as shown in Figure 5.4.

Users are supposed to avoid mentioning the attribute value they have in mind, but rather to provide the chatbot with indirect cues like *"I work in a* ***kitchen****"* for the *"Describe your working environment"* question. The number of questions is fixed to 5, which should provide enough cues in the user's utterances to predict the correct attribute value without a lengthy interaction. We also require that the user's response to a question contains at least four words.

To give users an idea of how responses should look, we provide a sample reply to each question, which the user can choose instead of typing their own responses. Each reply is

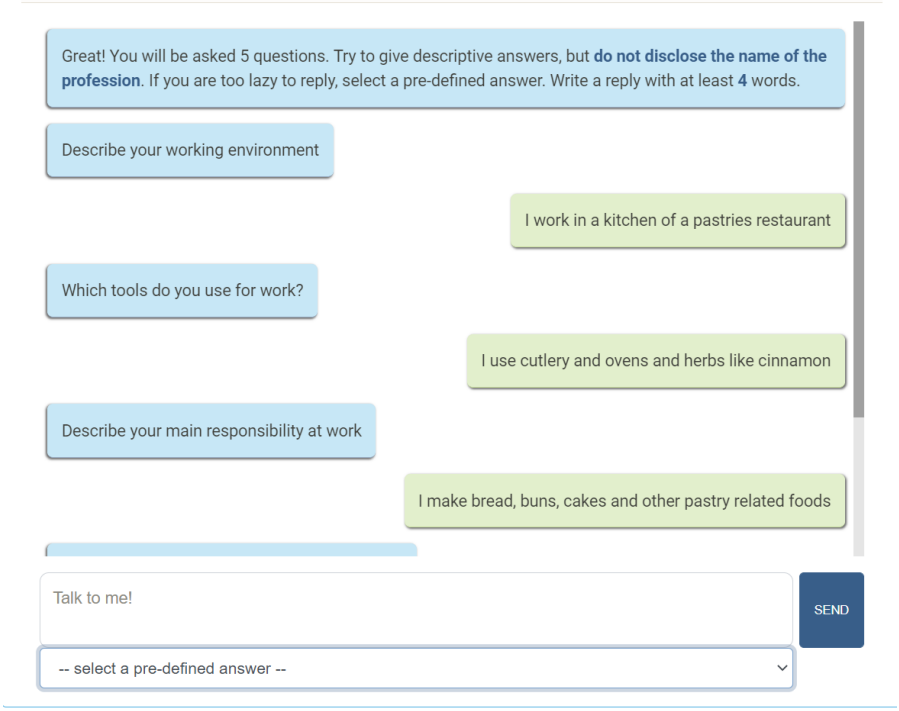


Figure 5.4: Chatbot conversation

designed as if it was given by a person with some pre-defined attribute value. For example, for the chat-bot request *"Describe the place where you do your hobby"*, we add a predefined reply *"It is a pool or open water"* related to *hobby:swimming*.

Social media setting. Social media traces of online users are utilized by large companies for personalizing their services and ads, making them more interesting and relevant. However, the users have neither control nor understanding of how their personal information was inferred and which parts of their content revealed it. Ideally, the users should be given an opportunity to identify and exclude their posts which can potentially expose personal facts.

In the social media scenario, we show how CHARM can dig through the vast amount of noisy conversational data in social media to find accurate cues for prediction. Users can type or paste their social media posts (e.g., Reddit submissions) into the social media interface of our demonstration platform. Together with CHARM's predictions, the users will be provided with the information which parts of their utterances were used by the predictive model. It provides an opportunity to delete or modify the exposing content, and to check whether the model can still arrive at the same prediction after a partial content removal.

As in the chatbot scenario, we provide samples of synthetic user-generated content, resembling submissions in Reddit discussion threads, corresponding to pre-defined attribute values.

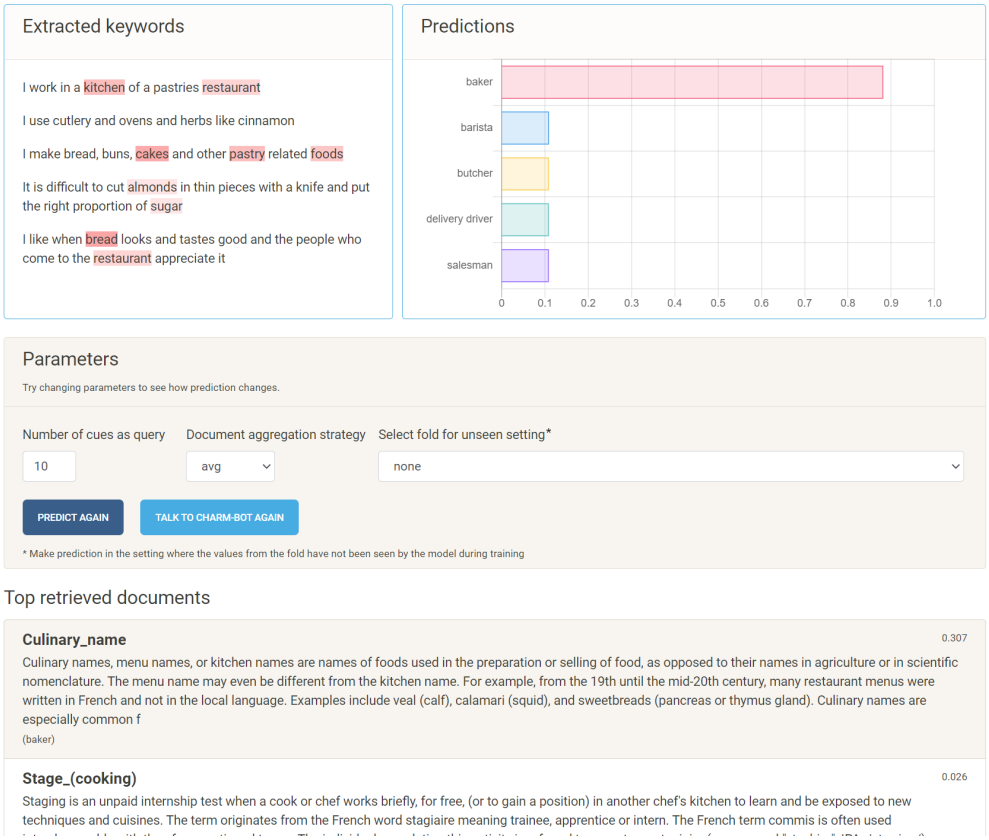


Figure 5.5: CHARM prediction result.

5.8.2.2 Prediction results

As shown in Figure 5.5, the prediction page presented to the user consists of intermediate results for both components of CHARM (*term scoring model* and *document ranker*) and the final prediction. To show the keyword selection step, we highlight the words in the user's utterances with an intensity that corresponds to the words' scores given by the term scoring model. The results for the document ranking step are presented as a sorted list of 10 top scoring documents. Each document is linked to the original article on the web. Finally, we show top 5 attribute value predictions after aggregating document scores. We normalized them to $[0, 1]$ scale for interpretability when comparing the model's confidence in predicting each value.

5.8.2.3 Model parameters

The demonstration allows the users to explore how CHARM's predictions change depending on its two hyperparameters: the *number of extracted keywords* and the *document aggregation strategy*. We set the default number of extracted keywords as $\frac{1}{3}$ of the number of meaningful terms in the input utterances (after removing stopwords and digits), with 10 as the maximum value. Setting the number of keywords too high can result in a noisy query and inadequate

behaviour of the retrieval model. On the other hand, small number of keywords can be insufficient for accurate document retrieval.

As the document aggregation strategy, the user can choose between `max` and `average` functions. `Max` operation is useful when the document collection is noisy and the prediction score should come from a single most relevant document per attribute value. The `average` function is good to provide a balanced prediction based on all available documents, protecting the result from being spoilt by an inappropriate top scoring document. For this reason we selected `average` as the default aggregation function.

The demonstration platform is implemented using Flask framework. We selected to use CHARM$_{\text{KNRM}}$ as the underlying model and Wiki-category as the document collection for our demonstration platform. As shown in the experiments described in Section 5.7.1, this combination of ranker and document collection shows superior performance in most test cases. Wiki-category is a sweet spot between simple Wiki-page, which can only provide trivial explanations with pages matching attribute value name, and Web-search collection, which is difficult for the end-user to interpret because of noisy and ill-formatted pages.

5.8.2.4 Unseen scenario

We also showcase CHARM's ability to predict attribute values that are lacking training samples. We train 10 variants of the model, in which each model has seen samples from only 90% of the attribute values during training; 10% of the attribute values are *unseen*. In our web interface, the users can try to make a prediction using one of those models by selecting the option where the listed attribute values are unseen. For example, for the input utterances {*"I was pedaling the whole evening"*, *"I don't like long walks, I like spending time on my bike"*}, it can be interesting to see the prediction result by the model not trained on hobby value *cycling*.

5.8.3 Case study

In this section we present a walk-though scenario for the chatbot setting. As input we take a set of utterances from a pre-defined personality having *profession*: *baker*. In the first step, the chatbot asks the user 5 questions, such as *"How do you start your day at work?"*. We give an excerpt of the conversation between the user and the chatbot in Figure 5.4.

On the next step the user is taken to the prediction result page, shown in Figure 5.5. Using the default heuristic, CHARM extracts 9 keywords from the input. The resulting query thus becomes *"kitchen restaurant cakes pastry foods almonds sugar bread restaurant"*. From Figure 5.5 it can be seen that the words '*cakes*', '*bread*' and '*pastry*' were assigned high scores by the term scoring model, whereas more general words, like '*restaurant*', were included in the query but received lower scores.

The default document score aggregation strategy is `average`, which helps to overcome the influence of the top scoring document *wiki:Stage_(cooking)* (a culinary internship), which was automatically labeled as *student*. Thus, if the user changes the aggregation function to `max`, the effect of document scoring makes *baker* and *student* almost equally probable.

The qualitative results of varying the parameters of CHARM on our exemplary input

number of keywords	aggregation strategy	seen/unseen setting	correct prediction score	best incorrect prediction score
10	avg	seen	0.91	0.19 (barista)
2	avg	seen	0.88	0.19 (sailor)
25	avg	seen	0.85	0.85 (barista)
10	**max**	seen	0.89	0.77 (student)
10	avg	**unseen**	0.91	0.32 (butcher)

Table 5.8: Prediction scores based on CHARM parameters.

are shown in Table 5.8. Setting the number of keywords to 2 still does not prevent CHARM from making a correct prediction using a concise query *"bread ovens"*. However, the model is not robust with a long query, resulting in the ranker yielding many documents equally relevant to this query, like *baker*, *barista* and *butcher* pages.

Finally, the user can inspect the behaviour of CHARM in the *unseen* setup, when the value *baker* was not present in the training data. To do that the user should select an unseen fold from the dropdown list, which contains the value *baker*. As shown in Table 5.8, CHARM is still capable of predicting the correct value. In contrast to the normal *seen* setting, the difference in scores for correct and incorrect predictions is less.

5.9 Conclusion

We presented the Conversational Hidden Attribute Retrieval Model (CHARM), a novel method for inferring personal traits from conversations. CHARM differs from prior work by its zero-shot ability to predict attribute values that are not present in the training samples at all.

We demonstrated the viability of CHARM for inferring users' unseen attribute values by comprehensive experiments with Reddit conversations on *profession* and *hobby* attributes, leveraging document collections from Wikipedia and web search results for CHARM's retrieval component. In the zero-shot setting CHARM shows significantly better performance than existing unsupervised keyword selectors, especially given the challenging conversation domain. Moreover, CHARM also performs on par with state-of-the-art fully supervised models in the regular classification setting.

CHARM is extensible to other long-tailed personal attributes, such as *favorite food type* or *preferred travel destination*, without changing the model's architecture or exhaustive manual effort to construct external document collections. Moreover, the components of CHARM, *term scoring model* and *retrieval model* are easily modifiable, allowing to plug in any emerging state-of-the-art architecture. Finally, the strength of CHARM is its end-to-end training, without any intermediate supervision steps, regardless of the absence of ground truth about the attribute values' keywords.

We have shown that CHARM's predictions are explainable by the keywords and documents it selects, which are sufficiently descriptive to enable CHARM to draw fine-grained distinction between similar attribute values. To showcase that, we created a web demonstra-

tion platform, enabling the users to interact with CHARM and explore its predictions. Such web service will be a helpful asset to provide the end users with transparent and explainable models.

As future work directions we see improving CHARM's performance in the seen setup and applying the model on further datasets, given the availability of the labeled samples. Moreover, as CHARM's ability to make predictions in the unseen setup heavily hinges on the external document collection, it is interesting to explore different sources and methods to collect the documents. As we have observed, both comprehensive and diverse collections (automatically created Web search collection) as well as highly precise collections with little noise (manually refined Wiki-category) can strengthen the performance on different attributes.

We envision a major extension of CHARM as the model, capable of predicting *open-ended* personal attributes, such as *favorite singer*. For such attributes it is impossible to create comprehensive lists of attribute values, especially given constantly emerging new entities. This problem can be tackled by means of zero-shot learning techniques with heavy reliance on external information sources, such as knowledge bases.

Chapter 6

Predicting Relationships in Dialogue Excerpts

Contents

Automatically extracted interpersonal relationships of conversation interlocutors can enrich personal knowledge bases to enhance personalized search, recommenders and chatbots. In this chapter we propose PRIDE: a neural multi-label classifier inferring speakers' relationships from conversations. PRIDE effectively utilizes the dialogue structure additionally augmenting it with external knowledge about speaker features and conversation style. Unlike prior works, we address multi-label prediction of fine-grained directed relationships. Extensive experiments on datasets based on screenplays of movies and TV series show superior performance of PRIDE compared to the state-of-the-art baselines.

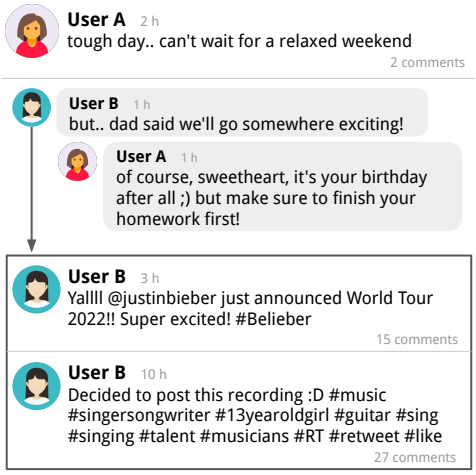


Figure 6.1: Example of two speaker conversation in social media.

6.1 Introduction

Motivation and Problem. A personal knowledge base enhanced with the information about the users' interpersonal relationships is practical for many applications. For example, relationship facts in a PKB can be accessed by a personalized chat-bot, which will enable it to make better suggestions for the user (for example, suggesting that the user takes her *child* to the zoo instead of a romantic dinner). Moreover, the speech style of the chat-bot, varying from neutral and official to casual and friendly, can be adjusted based on the relationship between the user and her current interlocutor. Finally, if the conversation happens over the phone, the underlying software can automatically assign categories (family/business/..) for the contact list of the user's interlocutors.

With the ubiquity of social media and online forums, user-generated content is available in abundance. Mining personal knowledge from user-generated content to populate PKBs, or *user profiling*, is a long-standing topic in NLP [9, 40, 152]. While users' demographic attributes and interests can be learned from their profile descriptions and posts, interpersonal relationships with other users are rarely mentioned explicitly and may only be inferred from their interactions and conversations.

In this work, we develop an automatic method for predicting fine-grained relationships between two speakers, given their logged conversation history.

Consider the example in Figure 6.1. From the excerpt of interactions between A and B, the reader can figure out that B is the *child* of A by observing *(i)* the address term '*sweetheart*', *(ii)* the commanding but soft tone of user A, *(iii)* the reference to the other family member '*dad*', and *(iv)* the context created by the word '*homework*'. Yet, neither of the speakers directly mentions their relationship, making this task difficult for automatic methods relying on explicit pattern matching or keyword search.

The relationship information extracted from such conversations, e.g., $\langle B,\ child\_of,\ A \rangle$, can be entered into the PKBs of users A and B. By combining such relationship information with User B's age and personal interests (e.g., *playing guitar, Justin Bieber*) inferable from

User B's social media (exemplified in Figure 6.1), a system will be able to provide user A with relevant personalized recommendations for a query "*birthday present ideas for my daughter*".

Prior Work and its Limitations. There has been considerable research on extracting relationships between characters in literary texts such as novels [16, 17]. These methods are inappropriate for conversational data, though, which is colloquial and less structured than literary texts. Moreover, predicting relationships is often modeled as a binary task of sentiment classification (i.e., person A is positive or negative about person B). Prior works on conversational data are restricted to small-scale data [178], or merely handle coarse labels of relationship aspects [125, 132]. Most approaches use general models for text classification [20, 59], which disregard the particularities of conversational settings.

Approach and Contributions. We present PRIDE, a neural multi-label classifier for **P**redicting **R**elationships **I**n **D**ialogu**E**. PRIDE makes inference among 12 fine-grained directed relationships (like *child* or *boss*) from conversational data by hierarchically creating utterance representations and combining them with signals on the users' personal attributes (e.g., *age* and *occupation*) and the conversation style (e.g., *intense* or *superficial*). PRIDE uses BERT [32] to create contextual word embeddings for each utterance, and Transformer encoders [158] to build conversation representations that preserve information about the sequence and speakers of utterances.

The contributions of this work are:

- a method for inferring speakers' relationships from conversational data, which outperforms strong baselines;
- an exhaustive analysis of the model's performance. We perform various experiments assessing PRIDE's transfer-learning capabilities and robustness to the varying lengths of the input conversations. Additionally, we conduct ablation studies, proving that all components of the model are essential for the accurate prediction of interpersonal relationships.

6.2 Related Work

The models HAM and CHARM, described in the previous two chapters, make predictions based on the input from a single speaker. Meanwhile, relationship inference requires processing the utterances of a pair of interlocutors in a conversation. In this section we will summarize related work on modeling multi-speaker dialogues. Many natural language processing tasks based on conversational speech (chatbot answer generation, utterance intent classification, emotion prediction, etc.) require creating a representation of a given multi-speaker conversation as input. We identify several features typical of the conversational data, which can be used to enhance predictive models: *(i)* conversational structure, *(ii)* speaker attribution, *(iii)* additional speaker information.

Conversational structure. One popular way to represent a conversation is to model words and utterances in a hierarchical manner. Hierarchical approaches are widely applied

to microblog sentiment and emotion classification. We gave a comprehensive overview of the hierarchical models in Section 4.2.2; in the current section we recap several recent methods, which inspired the choice of our model's architecture.

The core idea of hierarchical modeling is to create the representations of words, which are aggregated to create the representations for utterances; the latter are either used in the utterance-level inference or are further combined to make predictions on the conversational level.

A number of related studies use BERT to create the contextual representations of words, which are then processed by the recurrent models to form the representations of utterances [78, 98]. This approach is still not optimal because RNNs can not effectively capture the dependencies in the long input sequences and suffer from vanishing gradient. An alternative approach is to create utterance representations with Transformer [84, 144]; our proposed architecture also follows this approach. In contrast to prior works, the distinguishing feature of our method is an effective way to overcome the limitation on the number of BERT input tokens. As opposed to cropping or processing single utterances out of context [84, 144], we process the whole input conversations in large chuncks, joining them into a unified context with Transformer.

Another way to utilize dialogue structure is to use a graph to represent the conversation. Such approaches are used to process multi-party conversations involving more than two speakers, which often have non-sequential structure, as a single utterance can have multiple responses to it. An intuitive way to model such conversations is to use utterances as the vertices in a dialogue graph; an edge will then connect the response to its parent utterance [56]. Alternatively one can exploit a fully-connected graph [45], under the assumption that all utterances influence each other. Graph-based modelling has proven to be effective on natural language tasks such as emotion classification [45, 179]. However, it is unnecessary for out setting, as we consider only dyadic dialogues, modeling utterances' interactions with Transformer.

Speaker attribution. Speaker attribution (the information which speaker the current utterance was produced by) is often used across various NLP tasks to create speakers' representations. For example, in utterance addressee identification [75, 112] the models are trained to produce speakers' embeddings, which are explicitly used for addressee prediction. In other NLP tasks, such as sentiment classification or response selection, the learned speaker representations are blended into the model to enhance its performance.

To equip Transformer with speaker information, the studies by Liu et al. [91] and Li et al. [81] leverage specialized input masks to distinguish utterances from different speakers. These masks create distinct channels for each speaker in the encoder, so that an utterance representation can attend to the input from each speaker separately.

A simple but effective way to blend in speaker information into Transformer-based models, such as BERT, is to introduce additive speaker embeddings on the word level. For dyadic conversations, speakers are usually distinguished using BERT's segment embeddings [94]; for the conversations with more than two speakers a common solution is to add a separate speaker embedding layer into BERT's embedding module [49, 178].

PRIDE also incorporates speaker information; we add learned speaker embeddings to the Transformer input on the utterance level, following Li et al. [84], as well as using BERT's segment embeddings as speaker indicators on the word level.

Additional speaker information. Knowing the speaker of each utterance enables to link the available external knowledge about that speaker, making the model's predictions more accurate. There has been significant research on creating response selection models infused with pre-defined speaker personality [103, 183]. Such approaches operate on the utterance level, attaching personality to each generated utterance. As opposed to it, Welch et al. [165] enriched the model for speaker attribute prediction on the global (conversation) level, adding various features of the interlocutor, such as relative age or gender. Inspired by this approach, we also enhance our model with the information about the speakers' ages.

6.3 Background

Interactions between people can have multiple fine-grained features, describing various aspects of their communication. For example, interactions can be characterized by the attachment style (such as *commitment* or *avoidance*) [125] or power hierarchy (*subordinate* or *superior*) [121]. There are ample related social studies researching these characteristics; yet, there is no formal ontology for them [132]. One way to organize the features of interpersonal interactions was proposed by Rashid and Blanco [131], defined as *dimensions* of the relationships.

Most of the relationships that we defined for our experiments also have particular interpersonal characteristics. For example, the *enemy* relationship can be described as *competitive* as opposed to *cooperative*; the relationship between *parent* and *child* is in most cases *intimate*. Thus, we find it beneficial to enhance the relationship prediction model with the known features of the speakers' interactions.

We use the definition of *interpersonal dimensions* [169] of speakers' interactions and relationships, following classification by Rashid and Blanco [132], which we used as an additional input to our model. We note that the discussed interpersonal dimensions are descriptive of the relationship between a particular pair of speakers, but not of the relationship type in general (for example, the interaction between *colleagues* can be both *cooperative* and *competitive*). However, in general any interpersonal dimension can be fairly typical for a relationship type; we use this information to give the model hints about applicable predictions.

Rashid and Blanco consider 11 interpersonal dimensions, divided into dimensions of relationships and interactions, as shown in Table 6.1. In our model we use all proposed dimensions to provide a comprehensive summary of the relationship's fine-grain characteristics. Rashid and Blanco also provide a conversational dataset, where every utterance has annotations for each considered interpersonal dimension. We utilize this dataset to pretrain a model for utterance-level dimension classification and create separate representations for each dimension, which are later used in PRIDE.

relationships	cooperative vs. noncooperative pleasure vs. work oriented intimate vs.unintimate temporary vs. long term	equal vs. hierarchical intense vs. superficial active vs. passive
interactions	cooperative vs. noncooperative concurrent vs. non concurrent	active vs. passive near vs. distant

Table 6.1: Interpersonal dimensions used in PRIDE.

6.4 Methodology

The design of our model is based on conversational features such as dialogue structure (the order and boundaries of the utterances) and the attribution of each utterance to its corresponding speaker. The model architecture, inspired by Li et al. [84], is shown in Figure 6.2. PRIDE hierarchically creates word and utterance representations, which are then combined with representations of personal attributes and interpersonal dimensions (Table 6.1) to create a representation of the full conversation history. Given this representation of the conversation, a multi-label classification layer predicts one or more of the relationship labels, listed in Table 6.5. The model is trained with supervision on the relationship labels. In the following subsections we describe the model's components in more detail.

6.4.1 Contextual word representations

The input for a pair of speakers (sp_A, sp_B) is N utterances $u_1, ..., u_N$, where i-th utterance consists of words $w_i^1, ..., w_i^{n_i}$. In the first step, the word representations r_i^j are created with a function $f^{word}(w_1^1, ..., w_1^{n_1}, ..., w_N^{n_N}) = r_i^j$, which takes as input the concatenation of all utterances and produces the representations for each word. We chose BERT [32] to create word representations, because this model efficiently captures contextual information.

Considering that the maximal input length of BERT is 512 tokens, we split the input sequence of utterances into chunks and run BERT several times. Each chunk in the split has the maximal possible length that fits into one run without breaking individual utterances. We find this splitting strategy more effective than running BERT on single utterances [20] or short sequences which do not fully utilize max 512 limit [59]. In our method more conversational context is provided to create word representations. Also, simply truncating input to 512 tokens [94] might cause a loss of important cues.

As information about the current speaker we use BERT's segment embeddings, so that the A-segment corresponds to tokens from speaker A and the B-segment to speaker B. Furthermore, we encode the information about the utterance boundaries by prepending special tokens before each utterance: [s1] for the utterances of speaker A and [s2] for speaker B.

6.4.2 Utterance representations

Next, word representations r_i^j are aggregated within each utterance to create utterance representations r_i with the aggregation function $a^{word}(r_i^1, ..., r_i^{n_i}) = r_i$. The aggregation is

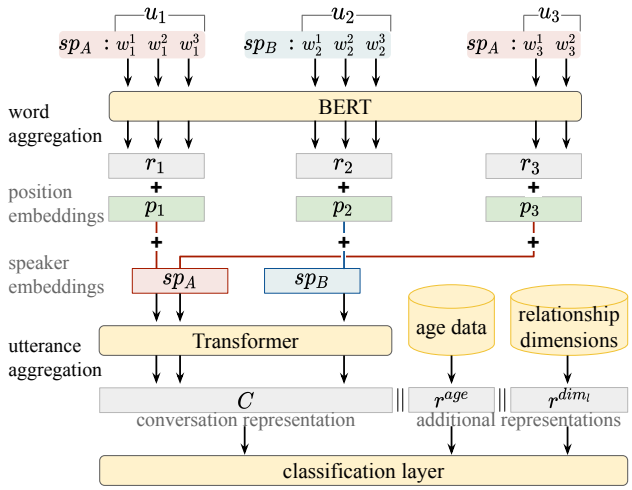


Figure 6.2: PRIDE model

performed on the utterances from all runs of BERT and outputs $r_1, ..., r_N$ as the representations of utterances. In our hyperparameter search we tried instantiating a^{word} with *max*, *average* and *self-attention weighted average* functions.

Some of r_i are produced by separate runs of BERT due to its input length limitation. Therefore we create enriched utterance representations in the unified context from all BERT runs with the function $f^{utt}(\hat{r}_1, ..., \hat{r}_n) = \tilde{r}_i$. We instantiate f^{utt} with a Transformer encoder, which allows us to input long sequences of utterances. Before computing enriched representations, we sum the utterance representations r_i with sinusoidal positional encoding p_i and speaker embeddings sp_i, yielding $\hat{r}_i = r_i + p_i + sp_i$. The speaker embeddings are randomly initialized and learned during model training. Positional encoding is performed following Vaswani et al. [158].

6.4.3 Classification layer

Finally, the enriched utterance representations $\tilde{r}_i$ are aggregated with the function $a^{utt}(\tilde{r_1}, ..., \tilde{r_n}) = C$. a^{utt} is instantiated with the same aggregation functions as a^{word}. For the case with [CLS] representation we prepend a trainable embedding to the sequence.

We incorporate additional information relevant to the relationship prediction by concatenating embeddings of personal attributes and interpersonal dimensions with the conversation representation C: $\tilde{C} = C|r^{age}|r_{dim_l}$, which are described in the following subsections. A fully connected layer takes the resulting concatenated representation $\tilde{C}$ as input and produces probability scores for each of L relationship labels. Since some relationships are not symmetric (e.g., *parent/child*) the labels represent directed relationships from sp_A to sp_B.

6.4.4 Incorporating personal attributes

Additional personal information about the speakers from a personal knowledge base, such as their *age* or *occupation*, could improve relationship prediction. We incorporate *age* information

into the model, since some relationships in our dataset can commonly be characterized by age differences between the speakers. For instance, children are usually much younger than their parents (and a parent can never be younger than a child). Similarly, employees are generally younger than their bosses (but the magnitude of their age difference is less than in parent/child pairs). Among all possible personal attributes we select to incorporate only age difference for two reasons: *(i)* other labeled personal attributes are very scarce and difficult to obtain, and *(ii)* other attributes, such as *gender*, are not nearly as informative as speakers' age difference. Nevertheless, the architecture of PRIDE can easily include any number of additional attributes (e.g., *profession*, *family status*, etc.).

To incorporate age information into PRIDE, we introduce a representation for the age difference of speakers. We calculate $(age_A - age_B)$ and assign the resulting difference into an age difference bin. We learn an m-dimensional embedding r^{age} for each bin, where m is a hyperparameter optimized in the grid search.

6.4.5 Incorporating interpersonal dimensions

As discussed in Section 6.3, interpersonal relationships have fine-grained characteristics, called *dimensions* [169]. For instance, a *boss/employee* relationship is hierarchical, while *colleague* is an equal one. Similarly, *spouse* is an intimate relationship, in contrast to *colleague*.

Given a hint of the applicable dimensions, a model can better predict the underlying relationship (e.g., a *non-intimate*, *task oriented* and *hierarchical* relationship is most likely a *boss/employee* relationship). Based on Rashid and Blanco [132], we distinguish 11 interesting interpersonal dimensions, listed in Table 6.1.

Using the data provided by Rashid and Blanco, we train a separate BERT classifier on the utterance level for each dimension dim_l, where index l ranges over the 11 interpersonal dimensions we used. We obtain a K-dimensional CLS representation from the trained classifier for each utterance, thus producing a K-dimensional representations $r_i^{dim_l}$ for the i-th input utterance. To incorporate these representations into our model, we obtain a single representation dim_l at the conversation level by performing max pooling over all utterance representations for a given speaker pair.

6.5 Experimental setup

6.5.1 Data splitting and preprocessing.

For experiments with PRIDE we used Film Relationship (FiRe) dataset, described in Section 3.2.3. From the input scripts we removed personal names[1] and movie-specific words (which we defined as words found in only one movie script), to reduce overfitting to movie domain or genre.

We performed five-fold cross-validation, training the models on three folds and choosing hyperparameter settings according to the performance on 1-fold validation set. We report the results on the remaining 1-fold test set. We arranged the folds so that the sets of movies, where the input character pairs come from, are disjoint. With that as a hard restriction,

[1] https://catalog.data.gov/dataset/baby-names-from-social-security-card-applications-national-data

we tried to maximally balance the label distributions across the folds. For that we created multiple random assignments of movies to folds and chose the one that maximized the balance metrics, which we defined as follows:

$$label\_balance = mean([\frac{d_l}{S_l} \text{ for l in labels}]),$$
$$d_l = \max_i s_l^i - \min_i s_l^i,$$

where S_l denotes the number of pairs for the label l in the whole dataset, and s_l^i denotes the number of pairs for the label l in fold i.

6.5.2 Model setup and evaluation metrics

We fine-tuned a pretrained BERT model model (*bert-base-uncased*) to create word embeddings. To produce interpersonal dimension embeddings, we train BERT on the labeled data from Rashid and Blanco [132] on each dimension separately, resulting in 768-dimensional representations.

We gathered the data about speakers' ages by crawling IMDb for the ages of the corresponding actors in the year the film/series was made. To create age embeddings we calculate the age difference (*diff*) between the speakers and assign it to one of the predefined *diff* bins. We set *diff* bins to be $[(-inf; -13], [-12; -6], [-5; -1], [0; 4], [5; 11], [12; +inf)]$ (a negative age difference means that speaker B is younger than speaker A).

Training mechanism. PRIDE is trained in two steps: first we train the model without external representations (age difference and interpersonal dimensions). Then the pretrained base model checkpoint is used in full PRIDE to train external representations' embeddings and classification layer (the weights of the base model stay frozen).

We trained the model with *binary cross-entropy* loss. During training we oversampled the under-represented labels. We perform grid search to tune the following hyperparameters: training epoch, learning rate (we use different learning rates for BERT and the rest of the model), word and utterance aggregation (among *max*, *average* and *attention*). We perform multi-label classification by predicting all labels with a score over a threshold, which we treat as a hyperparameter.

Evaluation metrics. We compute macro-averaged multilabel precision, recall and F1 score as evaluation metrics. During grid search we optimized F1 score of the performance on the development set.

6.5.3 Baselines.

We compare the performance of PRIDE with the following baselines:

- **RNN** is a BiLSTM [48] architecture adapted by Welch et al. [165], which was trained on short context windows. Before each utterance a special token ('⟨ME⟩' or '⟨OTHER⟩') is prepended to represent the speaker.

- **HAM**, described in Chapter 4. To allow multiple relationship predictions we trained $\text{HAM}_{\text{2attn}}$ for multi-label classification using binary cross-entropy loss. HAMs are designed

	cross-val on FiRe			train:FiRe, test:Series		
model	**F1**	**precision**	**recall**	**F1**	**precision**	**recall**
RNN	0.11	0.11	0.15	0.10	0.17	0.14
BERT$_{ddrel}$	0.20	0.20	0.25	0.14	0.22	0.15
HAM	0.23	0.25	0.22	0.16	0.21	0.16
BERT$_{conv}$	0.27	0.25	0.33	0.25	0.35	0.21
PRIDE	**0.38**	**0.42**	**0.37**	**0.30**	**0.43**	**0.29**

Table 6.2: Results on FiRe and Series datasets. The best scores (bold) significantly differ from the remaining ones measured by a McNemar's test ($p < 0.05$).

to process the utterances from a single speaker; for the experiments in this chapter we did not change the architecture of HAM$_{2\text{attn}}$, which was trained on the input from both speakers without incorporating any speaker information.

- **BERT**$_{conv}$ for sequence classification [94] runs on the concatenation of utterances divided by a [SEP] symbol and segment embeddings corresponding to the speaker of each utterance. The sequences of utterances greater than the allowed input length are cropped.
- **BERT**$_{ddrel}$ [59] produces the relationship label ranking for each dialogue snippet in a movie; the final scores for pair-level labels through the whole conversation history is the sum of MRRs of the labels from scenes' predictions.

PRIDE and all baselines are implemented using PyTorch. The code for all experiments is accessible at https://github.com/Anna146/PRIDE.

6.6 Results

6.6.1 Quantitative results

The main quantitative results are presented in Table 6.2. PRIDE outperforms all baselines by a large margin, including other BERT-based models. Unlike BERT$_{ddrel}$, which aggregates predictions on conversation snippets outside of the model, PRIDE internally learns the conversation representation. Furthermore, unlike BERT$_{conv}$, we do not crop the input sequence to 512 token limit and make use of the hierarchical structure of the conversations.

We also analyze PRIDE's transfer learning performance on the Series dataset as our test data. From the results shown in Table 6.2, we observe the same behaviour of the models, with PRIDE outperforming the baselines. F1 scores are generally lower than the evaluation on the FiRe dataset, due to the different nature of data (longer input sequences). PRIDE's precision is similar on both datasets, but the larger amount of input utterances with Series seem to reduce recall.

model	F1	precision	recall
RNN	0.04	0.02	0.10
BERT_{ddrel}	0.15	0.15	0.20
HAM	0.24	0.30	0.23
BERT_{conv}	0.23	0.32	0.23
PRIDE	**0.33**	**0.41**	**0.35**
human	0.84	0.89	0.79

Table 6.3: Results on a human-annotated FiRe subset.

model	F1	precision	recall
PRIDE	**0.38**	0.42	0.37
PRIDE − dimensions	0.36	0.36	0.40
PRIDE − age	0.37	0.38	0.37
PRIDE − speaker	0.35	0.37	0.36
PRIDE − positional	0.37	0.36	**0.41**
PRIDE − Transformer*	0.35	**0.46**	0.33

Table 6.4: Ablating elements of PRIDE. The models marked with * significantly differ with full PRIDE, measured by a McNemar's test ($p < 0.05$).

6.6.2 Comparison with human performance

It is often complicated even for humans to recognize the relationship between the speakers in a given conversation. Thus, human performance can be regarded as an upper bound on the model's performance. To obtain this upper bound estimation, we asked three human annotators to read the complete conversation history of two movie characters (the same as the input given to the model) and identify the applicable relationships. We sampled 5 pairs for each relationship label, resulting in 60 pairs. As human-predicted labels we assigned the relationships selected by at least 2 out of 3 annotators. The results on this dataset are shown in Table 6.3. While PRIDE substantially outperforms the baselines, it achieves about half of human precision, illustrating the difficulty of the given task.

6.6.3 Ablation study

To investigate the impact of different components of PRIDE on its performance, we run an ablation study, removing one PRIDE component at a time: we experimented on excluding additional age and interpersonal dimensions' representations as well as removing speaker and positional embeddings from Transformer's input. The ablation on Transformer is done by substituting it with aggregation operations on word and utterance levels consecutively. Results are shown in Table 6.4. It can be observed that removing positional encoding gives the least impact. On the other hand, the quality considerably drops by removing Transformer, which is caused by a very low recall. Removing other elements cause a drop in precision, suggesting that incorporating age differences and interpersonal dimensions improves performance.

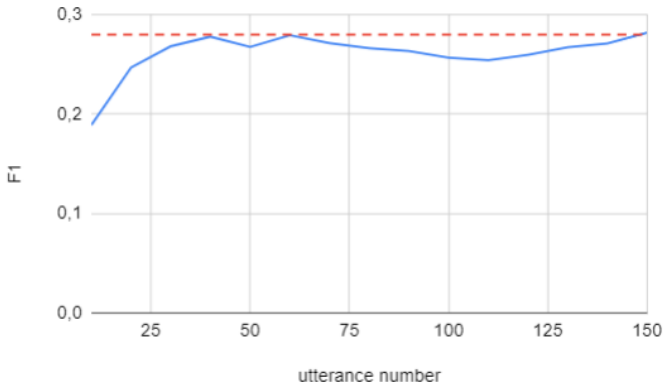


Figure 6.3: PRIDE F1 with varying input length. The dotted red line shows the performance on the full input.

class	count	PRIDE	(− speaker)	(− dimensions)
friend	208	0.50	0.50	0.50
lover	187	0.60	0.58	0.60
spouse	69	0.40	0.40	0.35
colleague	67	0.25	0.25	0.25
child	48	0.60	0.51	0.56
parent	41	0.62	0.55	0.60
sibling	37	0.42	0.33	0.40
employee	34	0.29	0.23	0.26
boss	29	0.04	0.08	0.04
enemy	27	0.14	0.13	0.14
medical	19	0.46	0.47	0.44
commercial	19	0.12	0.12	0.06

Table 6.5: Class F1 scores of PRIDE and PRIDE without speaker embeddings and interpersonal dimensions.

6.6.4 Varying input length

To investigate how many utterances are needed to make accurate predictions, we ran the trained PRIDE model on a subset of data with inputs of varying lengths. To do so, we selected a subset of user pairs with at least 150 utterances, and perform inference while increasing the length of the slice of input utterances from 10 to 150. This was repeated over 100 runs, with the randomized starting position of the slice. The results averaged over all runs are shown in Figure 6.3. We observe that approximately 40 utterances are enough to maximize performance in terms of F1 score.

6.6.5 Per class analysis

In Table 6.5 we show the label distribution and per class F1 scores for PRIDE and two ablated versions. We observe that using speaker embeddings benefit predictions on asymmetric classes, such as *child* and *parent*, as their F1 scores drop significantly when speaker embeddings are

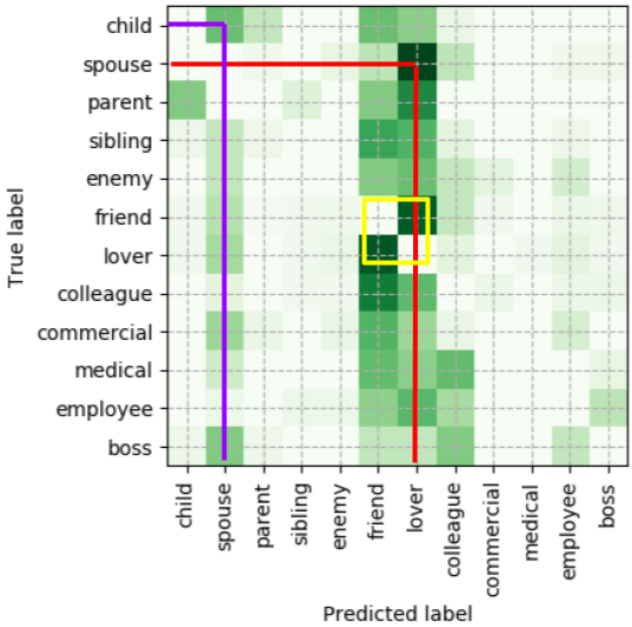


Figure 6.4: Confusion matrix for relationships.

not used. Removing interpersonal dimensions damages performance on *spouse* and *child* in particular, illustrating how this signal can help differentiate relationships that use similar vocabulary.

6.6.6 Misclassification analysis

The confusion matrix for PRIDE's predictions is shown in Figure 6.4. To create the confusion matrix for the multi-label case, we consider only incorrect predictions (either the labels which the model omitted or which it falsely predicted). For a single test instance we remove true positives from its sets of correct and predicted labels and use the Cartesian product of the resulting sets to build the matrix.

We observe that there are many misclassifications into the *friend* and *lover* labels, which are the most common (see columns). This can be attributed to the model's tendency to predict majority classes because of a considerable class imbalance.

Considering specific pairs, we see that the model often confuses *spouse* for *lover* (red line). They may talk to each other in a similar tone and use the same address terms. Conceptually, however, these classes are different, with spouses having tighter family bonds, discussing children and household issues, and lovers talking more casually. Similarly, *child* and *spouse* are often confused as well (purple line). Both may use terms related to family and discuss similar topics. The differences between *lover* and *friend* are indeed subtle (yellow square), and these pairs were also sometimes confused by human annotators.

Finally, we investigated the impact of confusion within asymmetric classes (for example, confusing *parent* to *child*). We found that if we accept the model's predictions of either label as correct, the average number of false positives for such classes drops by 34%, resulting in

an increase in average F1 score from 0.38 to 0.43. This illustrates the challenge posed by considering relationship directions and the importance of including asymmetric labels.

6.7 Conclusion

We presented PRIDE, a model for inferring fine-grained relationships from conversations. To our best knowledge, PRIDE is the first model to predict *directed, multilabel* speakers' relationships. PRIDE leverages the hierarchical dialogue structure to efficiently handle lengthy conversational history. The novelty of our architecture is the additional signals of speakers' demographics and speech style, which significantly improve relationship prediction.

PRIDE outperforms state-of-the-art baselines and demonstrates effective transfer learning on different types of dialogue data. PRIDE is designed to perform inference on long conversational sequences; however, we experimentally show PRIDE's ability to make accurate predictions for shorter interactions too.

To support future work on this topic, we created and released the largest labeled collection of relationships in conversations, which improves over existing datasets by including directed multilabel relationships.

6.7.1 Discussion

In this subsection we discuss several limitations of the current work and propose directions for further improvements of PRIDE:

- **Leveraging other types of conversational data.** Inferring relationships in real-life user conversations is the use case motivating our research. Thus, we find it important to evaluate PRIDE's transfer learning capabilities to other conversational datasets to ensure that it can generalize. Our choice of the dataset was constrained by the complexity of labeling dialogues with relationship labels; we leave it for future work to obtain more diverse relationship datasets (for example, social media interactions or telephone transcripts).

- **Improving performance on directed relationships.** Predicting asymmetric relationships has been overlooked in the prior works; yet accurately distinguishing them is important for practical applications. For instance, an intelligent assistant can recommend completely different items, depending of whether the user is asking for a birthday present suggestions for her *parent* or her *child*. Thus, we find it necessary to further improve PRIDE's performance on asymmetric relationships.

- **Incorporating more personal attributes.** In our experiments we showed that prediction of interpersonal relationships can benefit from adding speakers' attributes. We find it interesting to experiment on adding other personal information, such as *occupation* or *ethnicity*.

- **Joint prediction of personal attributes and interpersonal relationships.** The current version of PRIDE supports incorporating precomputed ground truth

information about the speakers' ages. In the scenario when personal attribute labels are not available, one option is to use a predictive model (such as HAM) to provide such information on the fly. Joint training of the relationship and speakers' attribute prediction models could improve their performance, as relationships and personal attributes are interdependent.

- **Considering multispeaker conversations.** The current dataset used in experiments with PRIDE was limited to uninterrupted dialogue spans between two characters. This limitation was due to the difficulty of distinguishing the addressee of an utterance when more than two speakers are present. In real life people often interact in a group, thus, considering only speaker pairs will result in losing useful cues for predictions. Therefore, extension of the current model to handle multi-speaker conversations should be further investigated.

Chapter 7

Conclusion

This thesis is concerned with predicting personal knowledge from conversations. Such information can be used to populate a personal knowledge base, enhancing many downstream applications. The ambiguity of conversational utterances makes it challenging to automatically process them; thus, the task of speakers' attribute inference is underexplored in related work. In our research we overcome the limitations of the prior studies, proposing the models which can accurately predict a wide range of personal facts.

In Chapter 4 we described *Hidden Attribute Models* (HAMs), capable of predicting speakers' demographic attributes: *age*, *gender*, *profession* and *family status*. HAMs utilize hierarchical conversational structure, making precise predictions at low computational costs. We have shown the capacity of HAMs to transfer learn among different conversational datasets, which is essential for applying the model to the real-life scenarios.

In Chapter 5 we presented *Conversational Hidden Attribute Retrieval Model* (CHARM), designed for predicting the values of the long-tailed *profession* and *hobby* attributes in a zero-shot setup. We propose a novel model design, which incorporates external knowledge to detect personal attribute values absent from the training data. CHARM makes predictions extracting keywords from the users' utterances, which ensure model's interpretability.

In Chapter 6 we introduced *PRIDE*, a model for *Predicting Relationships In Dialogue Excerpts*. Unlike most prior studies, PRIDE predicts fine-grained directed relationships, which are often ambiguous even for the human evaluators. We show that blending in additional signals, such as speakers' demographic attributes, can significantly improve interpersonal relationship inference.

Additionally, to support our experiments we issued several conversational datasets, described in Chapter 3. Our datasets cover multiple personal attributes, based on the dialogues in the movies and interactions on social media, providing diverse inputs for the models. Our labeling strategies and manual verification ensure high precision of the provided data, which will be valuable for further research and practical applications.

7.1 Future research directions

The research in this dissertation is only an initial step for a comprehensive and accurate prediction of personal information from conversations. In this section we list possible directions for further investigations, which we find essential for building practical and user-friendly personalized systems.

Open-ended attributes. Topical and user-oriented chat-bot recommendations require the knowledge of many open-ended personal attributes, e.g., *favorite actor*. It is infeasible to enumerate all possible values for such attributes, especially given that new values constantly emerge. Predicting such facts might require dedicated unsupervised extraction methods.

Continuous incremental predictions. An important aspect of conversational data is that the input utterances are spread in time, arriving as conversation proceeds. The utterances might contain contradictory cues, reflecting the change in the user's preferences (or even the user's demographics). Keeping an up-to-date state of the personal knowledge base as the conversation proceeds is an important issue, which can be addressed by learning personal facts from conversations incrementally.

Evaluating third party information. All our proposed models make predictions about a speaker (or a speaker pair) based on their own utterances. However, a significant amount of information can be obtained by capturing the input from other conversation participants. Ideally, the model should be able to capture both the cues from subject's direct interlocutor ("*you must be coming from your shift at the hospital*") and from a third person, when the subject is not even present in the current conversation ("*Brandon is doing a lot of overtime in the hospital recently*").

Utilizing speaker network. Building up on the previous point, we propose that the predictions of multiple personal attributes and relationships can be made simultaneously for a group of speakers, either within a current conversation or across multiple dialogues. A good example when such approach can facilitate predictions is utilizing the dependency of interpersonal relationships (from the fact that A and B are *children* of C one can infer that A and B are *siblings*). We envision that joint inference for all conversation participants can be performed with graph methods, which enable information sharing between the speakers (graph nodes).

Privacy issues. Personal attributes is a sensitive information, the exposure of which can be harmful for the end user. Our proposed models supply the evidence for their predictions, which provides the pointers to disclosing content of the user. We suggest that more research needs to be done into using this evidence to protect the users' personal data.

Bibliography

[1] Muhammad Ali, Piotr Sapiezynski, Miranda Bogen, Aleksandra Korolova, Alan Mislove, and Aaron Rieke. Discrimination through optimization: How facebook's ad delivery can lead to biased outcomes. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–30, 2019.

[2] Kaisheng Yao an. An attentional neural conversation model with improved specificity. *ArXiv preprint*, abs/1606.01292, 2016. URL https://arxiv.org/abs/1606.01292.

[3] Mahmoud Azab, Noriyuki Kojima, Jia Deng, and Rada Mihalcea. Representing movie characters in dialogues. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 99–109, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1010. URL https://aclanthology.org/K19-1010.

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *ArXiv preprint*, abs/1607.06450, 2016. URL https://arxiv.org/abs/1607.06450.

[5] Lei Jimmy Ba, Kevin Swersky, Sanja Fidler, and Ruslan Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 4247–4255. IEEE Computer Society, 2015. doi: 10.1109/ICCV.2015.483. URL https://doi.org/10.1109/ICCV.2015.483.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

[7] Krisztian Balog and Tom Kenter. Personal knowledge graphs: A research agenda. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 217–220, 2019.

[8] Krisztian Balog, Filip Radlinski, and Shushan Arakelyan. Transparent, scrutable and explainable user models for personalized recommendation. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 265–274. ACM, 2019. doi: 10.1145/3331184.3331211. URL https://doi.org/10.1145/3331184.3331211.

[9] Angelo Basile, Gareth Dwyer, Maria Medvedeva, Josine Rawee, Hessel Haagsma, and Malvina Nissim. N-GrAM: New Groningen Author-profiling Model—Notebook for

PAN at CLEF 2017. In *Working Notes Papers of the Conference and Labs of the Evaluation Forum, CLEF 2017 Evaluation Labs*, 2017.

[10] Roy Khristopher Bayot and Teresa Gonçalves. Age and gender classification of tweets using convolutional neural networks. In *Machine Learning, Optimization, and Big Data*, Cham, 2018. Springer International Publishing. ISBN 978-3-319-72926-8.

[11] Kalina Bontcheva and Dominic Rout. Making sense of social media streams through semantics: a survey. *Semantic Web*, 5(5):373–403, 2014.

[12] Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=S1Bb3D5gg.

[13] Adrien Bougouin, Florian Boudin, and Béatrice Daille. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan, 2013. Asian Federation of Natural Language Processing. URL https://aclanthology.org/I13-1062.

[14] John D. Burger, John Henderson, George Kim, and Guido Zarrella. Discriminating gender on Twitter. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1301–1309, Edinburgh, Scotland, UK., 2011. Association for Computational Linguistics. URL https://aclanthology.org/D11-1120.

[15] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. Yake! collection-independent automatic keyword extractor. In *European Conference on Information Retrieval*, pages 806–810. Springer, 2018.

[16] Snigdha Chaturvedi, Shashank Srivastava, Hal Daumé III, and Chris Dyer. Modeling evolving relationships between characters in literary novels. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2704–2710. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12408.

[17] Snigdha Chaturvedi, Mohit Iyyer, and Hal Daumé III. Unsupervised learning of evolving relationships between literary characters. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3159–3165. AAAI Press, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14564.

[18] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. Technical report, Google, 2013.

[19] Guandan Chen, Wenji Mao, Qingchao Kong, and Han Han. Joint learning with keyword extraction for event detection in social media. In *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 214–219. IEEE, 2018.

[20] Yi-Ting Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. MPDD: A multi-party dialogue dataset for analysis of emotions and interpersonal relationships. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 610–614, Marseille, France, 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.76.

[21] Yun-Nung Chen, Dilek Hakkani-Tür, Gokhan Tur, Jianfeng Gao, and Li Deng. End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding. In *Proceedings of Interspeech'16*, 2016. doi: 10.21437/Interspeech.2016-312.

[22] Wei Chu, Zoubin Ghahramani, and Christopher KI Williams. Gaussian processes for ordinal regression. *Journal of machine learning research*, 6(7), 2005.

[23] Christopher Cieri, David Miller, and Kevin Walker. The fisher corpus: a resource for the next generations of speech-to-text. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, 2004. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2004/pdf/767.pdf.

[24] Arman Cohan, Bart Desmet, Andrew Yates, Luca Soldaini, Sean MacAvaney, and Nazli Goharian. SMHD: a large-scale resource for exploring online language usage for multiple mental health conditions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1485–1497, Santa Fe, New Mexico, USA, 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1126.

[25] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

[26] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20 (3):273–297, 1995.

[27] Zhuyun Dai and Jamie Callan. Deeper text understanding for IR with contextual neural language modeling. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 985–988. ACM, 2019. doi: 10.1145/3331184.3331303. URL https://doi.org/10.1145/3331184.3331303.

[28] Cristian Danescu-Niculescu-Mizil and Lillian Lee. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*,

pages 76–87, Portland, Oregon, USA, 2011. Association for Computational Linguistics. URL https://aclanthology.org/W11-0609.

[29] Amit Datta, Michael Carl Tschantz, and Anupam Datta. Automated experiments on ad privacy settings. *Proceedings on Privacy Enhancing Technologies*, 2015(1):92–112, 2015.

[30] Yann N Dauphin, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. Zero-shot learning for semantic utterance classification. *ArXiv preprint*, abs/1401.0509, 2014. URL https://arxiv.org/abs/1401.0509.

[31] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the EM algorithm. *ArXiv preprint*, abs/10.2307, 2010. URL https://arxiv.org/abs/10.2307.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

[33] Brian S Everitt. *The analysis of contingency tables*. Chapman and Hall/CRC, 2019.

[34] Benjamin Fabian, Annika Baumann, and Marian Keil. Privacy on reddit? towards large-scale user classification. In *Proceedings of ECIS'15*, 2015.

[35] Zheng Fang, Yanan Cao, Qian Li, Dongjie Zhang, Zhenyu Zhang, and Yanbing Liu. Joint entity linking with deep reinforcement learning. In Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 438–447. ACM, 2019. doi: 10.1145/3308558.3313517. URL https://doi.org/10.1145/3308558.3313517.

[36] Ethan Fast, Binbin Chen, and Michael S. Bernstein. Empath: Understanding topic signals in large-scale text. In Jofish Kaye, Allison Druin, Cliff Lampe, Dan Morris, and Juan Pablo Hourcade, editors, *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, San Jose, CA, USA, May 7-12, 2016*, pages 4647–4657. ACM, 2016. doi: 10.1145/2858036.2858535. URL https://doi.org/10.1145/2858036.2858535.

[37] Jiazhan Feng, Chongyang Tao, Wei Wu, Yansong Feng, Dongyan Zhao, and Rui Yan. Learning a matching model with co-teaching for multi-turn response selection in retrieval-based dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3805–3815, Florence, Italy, 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1370. URL https://aclanthology.org/P19-1370.

[38] S Craig Finlay. Age and gender in reddit commenting and success. *Journal of Information Science Theory and Practice*, 2(3):18–28, 2014.

[39] Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[40] Lucie Flekova, Jordan Carpenter, Salvatore Giorgi, Lyle Ungar, and Daniel Preoţiuc-Pietro. Analyzing biases in human perception of user age and gender from text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 843–854, Berlin, Germany, 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1080. URL https://aclanthology.org/P16-1080.

[41] Francisco Manuel, Rangel Pardo, Paolo Rosso, Martin Potthast, and Benno Stein. Overview of the 5th Author Profiling Task at PAN 2017: Gender and Language Variety Identification in Twitter. In *Working Notes Papers of the Conference and Labs of the Evaluation Forum, CLEF 2017 Evaluation Labs*, 2017.

[42] Nikesh Garera and David Yarowsky. Modeling latent biographic attributes in conversational genres. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 710–718, Suntec, Singapore, 2009. Association for Computational Linguistics. URL https://aclanthology.org/P09-1080.

[43] Emma J Gerritse, Faegheh Hasibi, and Arjen P de Vries. Bias in conversational search: The double-edged sword of the personalized knowledge graph. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*, pages 133–136, 2020.

[44] Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. A knowledge-grounded neural conversation model. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5110–5117. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16710.

[45] Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. DialogueGCN: A graph convolutional neural network for emotion recognition in conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 154–164, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1015. URL https://aclanthology.org/D19-1015.

[46] Matej Gjurković and Jan Šnajder. Reddit: A gold mine for personality prediction. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 87–97, New Orleans, Louisiana,

USA, 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-1112. URL https://aclanthology.org/W18-1112.

[47] Philip John Gorinski and Mirella Lapata. What's this movie about? a joint neural network architecture for movie content analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1770–1781, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1160. URL https://aclanthology.org/N18-1160.

[48] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

[49] Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. Speaker-aware BERT for multi-turn response selection in retrieval-based chatbots. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux, editors, *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2041–2044. ACM, 2020. doi: 10.1145/3340531.3412330. URL https://doi.org/10.1145/3340531.3412330.

[50] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. *ArXiv preprint*, abs/2002.08909, 2020. URL https://arxiv.org/abs/2002.08909.

[51] Amelie Gyrard, Manas Gaur, Saeedeh Shekarpour, Krishnaprasad Thirunarayan, and Amit Sheth. Personalized health knowledge graph. In *First International Workshop on Contextualized Knowledge Graphs*, 2018.

[52] Christophe Van Gysel, Bhaskar Mitra, Matteo Venanzi, Roy Rosemarin, Grzegorz Kukla, Piotr Grudzien, and Nicola Cancedda. Reply with: Proactive recommendation of email attachments. In Ee-Peng Lim, Marianne Winslett, Mark Sanderson, Ada Wai-Chee Fu, Jimeng Sun, J. Shane Culpepper, Eric Lo, Joyce C. Ho, Debora Donato, Rakesh Agrawal, Yu Zheng, Carlos Castillo, Aixin Sun, Vincent S. Tseng, and Chenliang Li, editors, *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pages 327–336. ACM, 2017. doi: 10.1145/3132847.3132979. URL https://doi.org/10.1145/3132847.3132979.

[53] M. Habibi and A. Popescu-Belis. Keyword extraction and clustering for document recommendation in conversations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(4):746–759, 2015.

[54] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273, Baltimore, Maryland, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1119. URL https://aclanthology.org/P14-1119.

[55] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

[56] Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. GSN: A graph-structured network for multi-party dialogues. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5010–5016. ijcai.org, 2019. doi: 10.24963/ijcai.2019/696. URL https://doi.org/10.24963/ijcai.2019/696.

[57] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. Co-pacrr: A context-aware neural IR model for ad-hoc retrieval. In Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek, editors, *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages 279–287. ACM, 2018. doi: 10.1145/3159652.3159689. URL https://doi.org/10.1145/3159652.3159689.

[58] Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan Boyd-Graber, and Hal Daumé III. Feuding families and former Friends: Unsupervised learning for dynamic fictional relationships. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1534–1544, San Diego, California, 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1180. URL https://aclanthology.org/N16-1180.

[59] Qi Jia, Hongru Huang, and Kenny Q Zhu. Ddrel: A new dataset for interpersonal relation classification in dyadic dialogues. 35(14):13125–13133, 2021.

[60] Hongyan Jing, Nanda Kambhatla, and Salim Roukos. Extracting social networks and biographical facts from conversational speech transcripts. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 1040–1047, Prague, Czech Republic, 2007. Association for Computational Linguistics. URL https://aclanthology.org/P07-1131.

[61] Chaitanya K. Joshi, Fei Mi, and Boi Faltings. Personalization in goal-oriented dialog. In *Proceedings of Conversational AI Workshop, Neural Information Processing Systems, NIPS'17*, 2017.

[62] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1062. URL https://aclanthology.org/P14-1062.

[63] Kirthevasan Kandasamy, Yoram Bachrach, Ryota Tomioka, Daniel Tarlow, and David Carter. Batch policy gradient methods for improving neural conversation models.

In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rJfMusFll.

[64] Pei-Wei Kao, An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. Convlogminer: A real-time conversational lifelog miner. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, 2021.

[65] Pavan Kapanipathi, Prateek Jain, Chitra Venkataramani, and Amit Sheth. User interests identification on twitter using a hierarchical knowledge base. In *Proceedings of Extended Semantic Web Conference, ESWC'14*, pages 99–113. Springer, 2014.

[66] Denys Katerenchuk, David Guy Brizan, and Andrew Rosenberg. "was that your mother on the phone?": Classifying interpersonal relationships between dialog participants with lexical and acoustic properties. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[67] Evgeny Kim and Roman Klinger. Frowning Frodo, wincing Leia, and a seriously great friendship: Learning to classify emotional relationships of fictional characters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 647–653, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1067. URL https://aclanthology.org/N19-1067.

[68] Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, pages 619–627, 2012. URL http://proceedings.mlr.press/v22/kim12.html.

[69] Su Nam Kim and Timothy Baldwin. Extracting keywords from multi-party live chats. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*, pages 199–208, Bali, Indonesia, 2012. Faculty of Computer Science, Universitas Indonesia. URL https://aclanthology.org/Y12-1021.

[70] Sunghwan Mac Kim, Qiongkai Xu, Lizhen Qu, Stephen Wan, and Cécile Paris. Demographic inference on Twitter using recursive neural networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 471–477, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-2075. URL https://aclanthology.org/P17-2075.

[71] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://aclanthology.org/D14-1181.

[72] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[73] Bernhard Kratzwald and Stefan Feuerriegel. Adaptive document retrieval for deep question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 576–581, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1055. URL https://aclanthology.org/D18-1055.

[74] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. Zero-data learning of new tasks. In *AAAI*, volume 1, page 3, 2008.

[75] Ran Le, Wenpeng Hu, Mingyue Shang, Zhenjun You, Lidong Bing, Dongyan Zhao, and Rui Yan. Who is speaking to whom? learning to identify utterance addressee in multi-party conversations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1909–1919, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1199. URL https://aclanthology.org/D19-1199.

[76] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[77] Kyumin Lee, James Caverlee, and Steve Webb. The social honeypot project: protecting online communities from spammers. In Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti, editors, *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 1139–1140. ACM, 2010. doi: 10.1145/1772690.1772843. URL https://doi.org/10.1145/1772690.1772843.

[78] Jiahuan Lei, Qing Zhang, Jinshan Wang, and Hengliang Luo. BERT based hierarchical sequence classification for context-aware microblog sentiment analysis. In Tom Gedeon, Kok Wai Wong, and Minho Lee, editors, *Neural Information Processing*, ICONIP'19, pages 376–386, 2019. URL https://link.springer.com/chapter/10.1007%2F978-3-030-36718-3_32.

[79] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1034. URL https://aclanthology.org/K17-1034.

[80] Chenliang Li, Wei Zhou, Feng Ji, Yu Duan, and Haiqing Chen. A deep relevance model for zero-shot document filtering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2300–2310, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1214. URL https://aclanthology.org/P18-1214.

[81] Jiangnan Li, Zheng Lin, Peng Fu, Qingyi Si, and Weiping Wang. A hierarchical transformer with speaker modeling for emotion recognition in conversation. *ArXiv preprint*, abs/2012.14781, 2020. URL https://arxiv.org/abs/2012.14781.

[82] Jiwei Li, Alan Ritter, and Eduard Hovy. Weakly supervised user profile extraction from Twitter. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 165–174, Baltimore, Maryland, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1016. URL https://aclanthology.org/P14-1016.

[83] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany, 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1094. URL https://aclanthology.org/P16-1094.

[84] Qingbiao Li, Chunhua Wu, Zhe Wang, and Kangfeng Zheng. Hierarchical transformer network for utterance-level emotion recognition. *Applied Sciences*, 10(13), 2020. URL https://www.mdpi.com/2076-3417/10/13/4447.

[85] Xiang Li, Gökhan Tür, Dilek Z. Hakkani-Tür, and Qi Li. Personal knowledge graph population from user utterances in conversational understanding. In *Proceedings of IEEE Spoken Language Technology Workshop (SLT)*, 2014.

[86] Yumeng Li, Liang Yang, Bo Xu, Jian Wang, and Hongfei Lin. Improving user attribute classification with text and social network attention. *Cognitive Computation*, 11(4): 459–468, 2019.

[87] Andreas Liesenfeld, Gábor Parti, Yuyin Hsu, and Chu-Ren Huang. Predicting gender and age categories in English conversations using lexical, non-lexical, and turn-taking features. In *Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation*, pages 157–166, Hanoi, Vietnam, 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.paclic-1.19.

[88] Grace Lin and Marilyn Walker. All the world's a stage: Learning character models from film. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 6, 2011.

[89] Grace Lin and Marilyn Walker. All the world's a stage: Learning character models from film. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 6, 2011.

[90] Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 620–628, Boulder, Colorado, 2009. Association for Computational Linguistics. URL https://aclanthology.org/N09-1070.

[91] Longxiang Liu, Zhuosheng Zhang, Hai Zhao, Xi Zhou, and Xiang Zhou. Filling the gap of utterance-aware and speaker-aware representation for multi-turn dialogue. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*, 2021.

[92] Ye Liu, Sheng Zhang, Rui Song, Suo Feng, and Yanghua Xiao. Knowledge-guided open attribute value extraction with reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8595–8604, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.693. URL https://aclanthology.org/2020.emnlp-main.693.

[93] Robbie Love, Claire Dembry, Andrew Hardie, Vaclav Brezina, and Tony McEnery. The spoken bnc2014: Designing and building a spoken corpus of everyday conversations. *International Journal of Corpus Linguistics*, 22(3):319–344, 2017.

[94] Junyu Lu, Xiancong Ren, Yazhou Ren, Ao Liu, and Zenglin Xu. Improving contextual language models for response retrieval in multi-turn conversation. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 1805–1808. ACM, 2020. doi: 10.1145/3397271.3401255. URL https://doi.org/10.1145/3397271.3401255.

[95] Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. Learning personalized end-to-end goal-oriented dialog. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6794–6801. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33016794. URL https://doi.org/10.1609/aaai.v33i01.33016794.

[96] Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. Learning personalized end-to-end goal-oriented dialog. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6794–6801. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33016794. URL https://doi.org/10.1609/aaai.v33i01.33016794.

[97] Veronica Lynn, Niranjan Balasubramanian, and H. Andrew Schwartz. Hierarchical modeling for user personality prediction: The role of message-level attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5306–5316, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.472. URL https://aclanthology.org/2020.acl-main.472.

[98] Hui Ma, Jian Wang, Lingfei Qian, and Hongfei Lin. HAN-ReGRU: Hierarchical attention network with residual gated recurrent unit for emotion recognition in conversation. *Neural Computing and Applications*, 33:2685–2703, 2020. URL https://link.springer.com/article/10.1007%2Fs00521-020-05063-7.

[99] Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. Mem2Seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the*

56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1468–1478, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1136. URL https://aclanthology.org/P18-1136.

[100] Dejan Markovikj, S. Gievska, Michal Kosinski, and David Stillwell. Mining facebook data for predictive personality modeling. In *The International AAAI Conference on Web and Social Media, ICWSM 2013*, 2013.

[101] Philip Massey, Patrick Xia, David Bamman, and Noah A Smith. Annotating character relationships in literary texts. *ArXiv preprint*, abs/1512.00728, 2015. URL https://arxiv.org/abs/1512.00728.

[102] Yutaka Matsuo and Mitsuru Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(01):157–169, 2004.

[103] Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1298. URL https://aclanthology.org/D18-1298.

[104] Peter McCullagh and John A Nelder. *Generalized linear models*. Routledge, 2019.

[105] Matthew Michelson and Sofus A Macskassy. Discovering users' topics of interest on twitter: a first look. In *Proceedings of the fourth workshop on Analytics for noisy unstructured text data*, pages 73–80, 2010.

[106] Rada Mihalcea and Paul Tarau. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-3252.

[107] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *ArXiv preprint*, abs/1301.3781, 2013. URL https://arxiv.org/abs/1301.3781.

[108] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html.

[109] Kaixiang Mo, Yu Zhang, Shuangyin Li, Jiajun Li, and Qiang Yang. Personalizing a dialogue system with transfer reinforcement learning. In Sheila A. McIlraith and

Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5317–5324. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16104.

[110] Eric T. Nalisnick and Henry S. Baird. Character-to-character sentiment analysis in shakespeare's plays. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 479–483, Sofia, Bulgaria, 2013. Association for Computational Linguistics. URL https://aclanthology.org/P13-2085.

[111] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *ArXiv preprint*, abs/1901.04085, 2019. URL https://arxiv.org/abs/1901.04085.

[112] Hiroki Ouchi and Yuta Tsuboi. Addressee and response selection for multi-party conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2133–2143, Austin, Texas, 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1231. URL https://aclanthology.org/D16-1231.

[113] Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, and Tom M. Mitchell. Zero-shot learning with semantic output codes. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, pages 1410–1418. Curran Associates, Inc., 2009. URL https://proceedings.neurips.cc/paper/2009/hash/1543843a4723ed2ab08e18053ae6dc5b-Abstract.html.

[114] Panupong Pasupat and Percy Liang. Zero-shot entity extraction from web pages. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 391–401, Baltimore, Maryland, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1037. URL https://aclanthology.org/P14-1037.

[115] Eyal Peer, Laura Brandimarte, Sonam Samat, and Alessandro Acquisti. Beyond the turk: Alternative platforms for crowdsourcing behavioral research. *Journal of Experimental Social Psychology*, 70:153–163, 2017.

[116] Marco Pennacchiotti and Ana-Maria Popescu. A machine learning approach to twitter user classification. In *Fifth International AAAI Conference on Weblogs and Social Media*, 2011.

[117] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

[118] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

[119] Guangyuan Piao and John G Breslin. Inferring user interests in microblogging social networks: a survey. *User Modeling and User-Adapted Interaction*, 28(3):277–329, 2018.

[120] Martin Potthast, Francisco Rangel, Michael Tschuggnall, Efstathios Stamatatos, Paolo Rosso, and Benno Stein. Overview of PAN'17: Author Identification, Author Profiling, and Author Obfuscation. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. 7th International Conference of the CLEF Initiative (Conference and Labs of the Evaluation Forum, CLEF 17)*, 2017.

[121] Vinodkumar Prabhakaran and Owen Rambow. Predicting power relations between participants in written dialog from a single thread. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 339–344, Baltimore, Maryland, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2056. URL https://aclanthology.org/P14-2056.

[122] Daniel Preoţiuc-Pietro and Lyle Ungar. User-level race and ethnicity predictors from Twitter text. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1534–1545, Santa Fe, New Mexico, USA, 2018. Association for Computational Linguistics. URL https://aclanthology.org/C18-1130.

[123] Daniel Preoţiuc-Pietro, Vasileios Lampos, and Nikolaos Aletras. An analysis of the user occupational class through Twitter content. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1754–1764, Beijing, China, 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1169. URL https://aclanthology.org/P15-1169.

[124] Daniel Preoţiuc-Pietro, Ye Liu, Daniel Hopkins, and Lyle Ungar. Beyond binary labels: Political ideology prediction of Twitter users. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 729–740, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1068. URL https://aclanthology.org/P17-1068.

[125] Saira Qamar, Hasan Mujtaba, Hammad Majeed, and Mirza Omer Beg. Relationship identification between conversational agents using emotion analysis. *Cognitive Computation*, 13:673–687, 2021. URL https://link.springer.com/article/10.1007/s12559-020-09806-5.

[126] Kun Qian and Zhou Yu. Domain adaptive dialog generation via meta learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2639–2649, 2019.

[127] Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W. Bruce Croft. Learning to selectively transfer: Reinforced transfer learning for deep text matching. In J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman, editors, *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 699–707. ACM, 2019. doi: 10.1145/3289600.3290978. URL https://doi.org/10.1145/3289600.3290978.

[128] Mandyam Annasamy Raghuram, K Akshay, and K Chandrasekaran. Efficient user profiling in twitter social network using traditional classifiers. In *Intelligent systems technologies and applications*, pages 399–411. Springer, 2016.

[129] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ, 2003.

[130] Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. Classifying latent user attributes in twitter. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pages 37–44. ACM, 2010.

[131] Farzana Rashid and Eduardo Blanco. Dimensions of interpersonal relationships: Corpus and experiments. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2307–2316, Copenhagen, Denmark, 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1244. URL https://aclanthology.org/D17-1244.

[132] Farzana Rashid and Eduardo Blanco. Characterizing interactions and relationships between people. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4395–4404, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1470. URL https://aclanthology.org/D18-1470.

[133] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access, 2017.

[134] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009. ISSN 1554-0669.

[135] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109, 1995.

[136] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1:1–20, 2010.

[137] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv preprint*, abs/1910.01108, 2019. URL https://arxiv.org/abs/1910.01108.

[138] Maarten Sap, Gregory Park, Johannes Eichstaedt, Margaret Kern, David Stillwell, Michal Kosinski, Lyle Ungar, and Hansen Andrew Schwartz. Developing age and gender predictive lexica over social media. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1146–1151, Doha, Qatar, 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1121. URL https://aclanthology.org/D14-1121.

[139] Nicolas Schrading, Cecilia Ovesdotter Alm, Ray Ptucha, and Christopher Homan. An analysis of domestic abuse discourse on Reddit. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2577–2583, Lisbon, Portugal, 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1309. URL https://aclanthology.org/D15-1309.

[140] H. Andrew Schwartz, Johannes C. Eichstaedt, Margaret L. Kern, Lukasz Dziurzynski, Stephanie Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin E. P. Seligman, and Lyle H. Ungar. Personality, gender, and age in the language of social media: The open-vocabulary approach. In *Public Library of Science, PloS one*, 2013.

[141] H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. Personality, gender, and age in the language of social media: The open-vocabulary approach. *Public Library of Science, PloS one*, 8(9): e73791, 2013.

[142] Nacéra Bennacer Seghouani, Coriane Nana Jipmo, and Gianluca Quercini. Determining the interests of social media users: two approaches. *Information Retrieval Journal*, 22 (1-2):129–158, 2019.

[143] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11957.

[144] Yong Shan, Zekang Li, Jinchao Zhang, Fandong Meng, Yang Feng, Cheng Niu, and Jie Zhou. A contextual hierarchical attention network with adaptive objective for dialogue state tracking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6322–6333, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.563. URL https://aclanthology.org/2020.acl-main.563.

[145] Judy Hanwen Shen and Frank Rudzicz. Detecting anxiety through Reddit. In *Proceedings of the Fourth Workshop on Computational Linguistics and Clinical Psychology — From Linguistic Signal to Clinical Reality*, pages 58–65, Vancouver, BC, 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-3107. URL https://aclanthology.org/W17-3107.

[146] Luke Sloan, Jeffrey Morgan, Pete Burnap, and Matthew Williams. Who tweets? deriving the demographic characteristics of age, occupation and social class from twitter user meta-data. *Public Library of Science, PloS one*, 10(3):e0115545, 2015.

[147] Yiping Song, Zequn Liu, Wei Bi, Rui Yan, and Ming Zhang. Learning to customize language model for generation-based dialog systems. *ArXiv preprint*, abs/1910.14326, 2019. URL https://arxiv.org/abs/1910.14326.

[148] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205, Denver, Colorado, 2015. Association for Computational Linguistics. doi: 10.3115/v1/N15-1020. URL https://aclanthology.org/N15-1020.

[149] Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. Deep semantic role labeling with self-attention. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4929–4936. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16725.

[150] Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. Why self-attention? a targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1458. URL https://aclanthology.org/D18-1458.

[151] Mike Thelwall and Emma Stuart. She's reddit: A source of statistically significant gendered interest information? *Information Processing Management*, 56(4):1543 – 1558, 2019. ISSN 0306-4573. doi: https://doi.org/10.1016/j.ipm.2018.10.007. URL http://www.sciencedirect.com/science/article/pii/S0306457318304692.

[152] Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. Listening between the lines: Learning personal attributes from conversations. In Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 1818–1828. ACM, 2019. doi: 10.1145/3308558.3313498. URL https://doi.org/10.1145/3308558.3313498.

[153] Anna Tigunova, Paramita Mirza, Andrew Yates, and Gerhard Weikum. RedDust: a large reusable dataset of Reddit user traits. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6118–6126, Marseille, France, 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.751.

[154] Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. CHARM: Inferring personal attributes from conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5391–5404, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.434. URL https://aclanthology.org/2020.emnlp-main.434.

[155] Anna Tigunova, Paramita Mirza, Andrew Yates, and Gerhard Weikum. Exploring personal knowledge extraction from conversations with charm. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 1077–1080, 2021.

[156] Lingraj S Vannur, Balaji Ganesan, Lokesh Nagalapatti, Hima Patel, and MN Thippeswamy. Data augmentation for personal knowledge base population. *ArXiv preprint*, abs/2002.10943, 2020. URL https://arxiv.org/abs/2002.10943.

[157] Evgenii Vasilev. Inferring gender of reddit users. Master's thesis, Universität Koblenz-Landau, Universitätsbibliothek, 2018.

[158] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[159] Prashanth Vijayaraghavan, Soroush Vosoughi, and Deb Roy. Twitter demographic classification using deep multi-modal multi-task learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 478–483, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-2076. URL https://aclanthology.org/P17-2076.

[160] Byron C. Wallace, Do Kook Choe, Laura Kertz, and Eugene Charniak. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 512–516, Baltimore, Maryland, 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2084. URL https://aclanthology.org/P14-2084.

[161] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. R 3: Reinforced ranker-reader for open-domain question answering. In *Proceedings of AAAI'18*, 2018.

[162] Tingting Wang, Jie Zhou, Qinmin Vivian Hu, and Liang He. Aspect-level sentiment classification with reinforcement learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[163] Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–37, 2019.

[164] Zhilin Wang, Xuhui Zhou, Rik Koncel-Kedziorski, Alex Marin, and Fei Xia. Extracting and inferring personal attributes from dialogue. *arXiv preprint arXiv:2109.12702*, 2021.

[165] Charles Welch, Verónica Pérez-Rosas, Jonathan K Kummerfeld, and Rada Mihalcea. Look who's talking: Inferring speaker attributes from personal longitudinal dialog. *ArXiv preprint*, abs/1904.11610, 2019. URL https://arxiv.org/abs/1904.11610.

[166] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. Twitterrank: finding topic-sensitive influential twitterers. In Brian D. Davison, Torsten Suel, Nick Craswell, and Bing Liu, editors, *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010*, pages 261–270. ACM, 2010. doi: 10.1145/1718487.1718520. URL https://doi.org/10.1145/1718487.1718520.

[167] Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier R. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, pages 2035–2043. Curran Associates, Inc., 2009. URL https://proceedings.neurips.cc/paper/2009/hash/f899139df5e1059396431415e770c6dd-Abstract.html.

[168] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[169] Myron Wish, Morton Deutsch, and Susan J Kaplan. Perceived dimensions of interpersonal relations. *Journal of Personality and social Psychology*, 33(4):409–420, 1976. URL https://www.sciencedirect.com/science/article/pii/B9780080237190500176.

[170] Chien-Sheng Wu, Andrea Madotto, Zhaojiang Lin, Peng Xu, and Pascale Fung. Getting to know you: User attribute extraction from dialogues. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 581–589, Marseille, France, 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.73.

[171] Wei Wu, Bin Zhang, and Mari Ostendorf. Automatic generation of personalized annotation tags for Twitter users. In *Human Language Technologies: The 2010 Annual*

Conference of the North American Chapter of the Association for Computational Linguistics, pages 689–692, Los Angeles, California, 2010. Association for Computational Linguistics. URL https://aclanthology.org/N10-1101.

[172] Chen Xing, Yu Wu, Wei Wu, Yalou Huang, and Ming Zhou. Hierarchical recurrent attention network for response generation. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5610–5617. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16510.

[173] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. End-to-end neural ad-hoc ranking with kernel pooling. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 55–64. ACM, 2017. doi: 10.1145/3077136.3080809. URL https://doi.org/10.1145/3077136.3080809.

[174] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California, 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1174. URL https://aclanthology.org/N16-1174.

[175] Majid Yazdani and James Henderson. A model of zero-shot learning of spoken language understanding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 244–249, Lisbon, Portugal, 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1027. URL https://aclanthology.org/D15-1027.

[176] An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. Personal knowledge base construction from text-based lifelogs. In Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer, editors, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 185–194. ACM, 2019. doi: 10.1145/3331184.3331209. URL https://doi.org/10.1145/3331184.3331209.

[177] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4:259–272, 2016. doi: 10.1162/tacl_a_00097. URL https://aclanthology.org/Q16-1019.

[178] Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. Dialogue-based relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4927–4940, Online, 2020. Association for Computational Linguistics.

doi: 10.18653/v1/2020.acl-main.444. URL https://aclanthology.org/2020.acl-main.444.

[179] Dong Zhang, Liangqing Wu, Changlong Sun, Shoushan Li, Qiaoming Zhu, and Guodong Zhou. Modeling both context- and speaker-sensitive dependence for emotion detection in multi-speaker conversations. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 5415–5421. ijcai.org, 2019. doi: 10.24963/ijcai.2019/752. URL https://doi.org/10.24963/ijcai.2019/752.

[180] Hainan Zhang, Yanyan Lan, Liang Pang, Jiafeng Guo, and Xueqi Cheng. ReCoSa: Detecting the relevant contexts with self-attention for multi-turn dialogue generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3721–3730, Florence, Italy, 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1362. URL https://aclanthology.org/P19-1362.

[181] Jingqing Zhang, Piyawat Lertvittayakumjorn, and Yike Guo. Integrating semantic knowledge to tackle zero-shot text classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1031–1040, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1108. URL https://aclanthology.org/N19-1108.

[182] Qi Zhang, Yang Wang, Yeyun Gong, and Xuanjing Huang. Keyphrase extraction using deep recurrent neural networks on Twitter. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 836–845, Austin, Texas, 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1080. URL https://aclanthology.org/D16-1080.

[183] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1205. URL https://aclanthology.org/P18-1205.

[184] Tianyang Zhang, Minlie Huang, and Li Zhao. Learning structured representation for text classification via reinforcement learning. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 6053–6060. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16537.

[185] Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achanauparp, Ee-Peng Lim, and Xiaoming Li. Topical keyphrase extraction from Twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human*

Language Technologies, pages 379–388, Portland, Oregon, USA, 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1039.

[186] Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. Personalized dialogue generation with diversified traits. *ArXiv preprint*, abs/1901.09672, 2019. URL https://arxiv.org/abs/1901.09672.

[187] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the Very Large Data Base, VLDB Endowment*, 10(5):541–552, 2017.

[188] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany, 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-2034. URL https://aclanthology.org/P16-2034.

List of Figures

List of Tables

External Tools and Datasets

BERT model contextualized word embeddings https://huggingface.co/transformers/model_-doc/bert.html. 69, 91

Cornell Movie-Dialogs Corpus http://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html. 25

Crowdsourcing benchmark https://zhydhkcws.github.io/crowd_truth_infer-ence/index.html. 27

ELMo model bidirectional embedding model https://allennlp.org/elmo. 52

Empath text analysis across lexical categories https://github.com/Ejhfast/empath-client. 37

Flask micro web framework https://flask.palletsprojects.com. 79

GloVe word embeddings https://nlp.stanford.edu/projects/glove. 52

GradeSaver movie summaries https://www.gradesaver.com. 27

IMDb internet movie database https://www.imdb.com. 25, 91

IMSDb internet movie scrips database https://imsdb.com. 28

MTurk crowsourcing platform https://www.mturk.com. 22, 46

PersonaChat corpus crowdourced conversational corpus http://convai.io/#personachat-convai2-dataset. 46

PyTorch deep learning library https://pytorch.org. 49, 69, 92

RAKE keyword extraction method https://pypi.org/project/rake-nltk. 69

Reddit dump https://files.pushshift.io/reddit. 30

Scikit-learn machine learning library https://scikit-learn.org. 49

Snorkel framework weak supervision data labeling https://www.snorkel.org. 36

TextRank keyword extraction method https://pypi.org/project/pytextrank. 69

TREC Web Track text retrieval task https://trec.nist.gov/data/webmain.html. 69

word2vec word embeddings https://code.google.com/archive/p/word2vec. 47, 48, 52