Hybrid Machine Translation using Binary Classification Models trained on Joint, Binarised Feature Vectors

---

Thesis for obtaining the title of
Doctor of Engineering Science
of the Faculty of Natural Science and Technology I
of Saarland University

by

Dipl.-Inform. Christian Federmann

Saarbrücken
November 2013

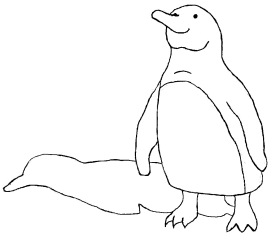
UNIVERSITÄT
DES
SAARLANDES

Day of Colloquium	December 16, 2013
Dean of the Faculty	Prof. Dr. Mark Groves
Chair of the Committee	Prof. Dr. Reinhard Wilhelm
First reviewer	Prof. Dr. Hans Uszkoreit
Second reviewer	Prof. Dr. Josef van Genabith
Academic Assistant	Dr. Hans-Ulrich Krieger

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Saarbrücken, November 2013



in memoriam:
Dr. Gabriele Federmann (1954–2013)

"The dead don't die. They look on and help."

– D. H. Lawrence: in a letter to J. Middleton Murray, February 2nd, 1923.

Abstract

We describe the design and implementation of a system combination method for machine translation output. It is based on sentence selection using binary classification models estimated on joint, binarised feature vectors. By contrast to existing system combination methods which work by dividing candidate translations into n-grams, i.e., sequences of n words or tokens, our framework performs sentence selection which does not alter the selected, best translation. First, we investigate the potential performance gain attainable by optimal sentence selection. To do so, we conduct the largest meta-study on data released by the yearly Workshop on Statistical Machine Translation (WMT). Second, we introduce so-called joint, binarised feature vectors which explicitly model feature value comparison for two systems A, B. We compare different settings for training binary classifiers using single, joint, as well as joint, binarised feature vectors. After having shown the potential of both selection and binarisation as methodological paradigms, we combine these two into a combination framework which applies pairwise comparison of all candidate systems to determine the best translation for each individual sentence. Our system is able to outperform other state-of-the-art system combination approaches; this is confirmed by our experiments. We conclude by summarising the main findings and contributions of our thesis and by giving an outlook to future research directions.

Zusammenfassung

Wir beschreiben den Entwurf und die Implementierung eines Systems zur Kombination von Übersetzungen auf Basis nicht modifizierender Auswahl gegebener Kandidaten. Die zugehörigen, binären Klassifikationsmodelle werden unter Verwendung von gemeinsamen, binärisierten Merkmalsvektoren trainiert. Im Gegensatz zu anderen Methoden zur Systemkombination, die die gegebenen Kandidatenübersetzungen in n-Gramme, d.h., Sequenzen von n Worten oder Symbolen zerlegen, funktioniert unser Ansatz mit Hilfe von nicht modifizierender Auswahl der besten Übersetzung. Zuerst untersuchen wir das Potenzial eines solches Ansatzes im Hinblick auf die maximale theoretisch mögliche Verbesserung und führen die größte Meta-Studie auf Daten, welche jährlich im Rahmen der Arbeitstreffen zur Statistischen Maschinellen Übersetzung (WMT) veröffentlicht worden sind, durch. Danach definieren wir sogenannte gemeinsame, binärisierte Merkmalsvektoren, welche explizit den Merkmalsvergleich zweier Systeme A, B modellieren. Wir vergleichen verschiedene Konfigurationen zum Training binärer Klassifikationsmodelle basierend auf einfachen, gemeinsamen, sowie gemeinsamen, binärisierten Merkmalsvektoren. Abschließend kombinieren wir beide Verfahren zu einer Methodik, die paarweise Vergleiche aller Quellsysteme zur Bestimmung der besten Übesetzung einsetzt. Wir schließen mit einer Zusammenfassung und einem Ausblick auf zukünftige Forschungsthemen.

Acknowledgments

Writing these final words of my thesis happens as a strange and yet marvelous year comes to a close. I find myself living in Vancouver, working for Microsoft Research and suddenly faced with the task to help improving a machine translation system used on a global scale. Challenge accepted.

I would like to thank my wife Kira for her patience and support during the writing of this document. I am looking forward to all the fun things we will experience in the Pacific North West. Thanks for sharing this with me!

This work is dedicated to the memory of my mother who passed away unexpectedly earlier this year. I also want to thank my sister Maike, my brother Alex, my dad and my grandmother for supporting me over the years. Don't fear, I will not get lost over here!

I would like to thank my supervisor, Hans Uszkoreit, for his support and encouragement which allowed me to pursue the research reported on in my thesis. After having worked with Hans for over ten years, I am grateful for his mentorship and friendship.

Likewise, I am indebted to my second supervisor, Josef van Genabith, for his insightful comments and inspiration along the way. Thank you very much for that, Josef!

Over the years, I had the good fortune to work with a bunch of great people. Thanks (in no particular order) go to Andreas Eisele, Marc Schröder, Hans-Ulrich Krieger, Bernd Kiefer, Hendrik Zender, Geert-Jan Kruijff, Ivana Kruijff-Korbayova, Thierry Declerck, Yu Chen, Yi Zhang, Miroslav Janicek, Corinna Johanns, Alexandre Klementiev, Günter Neumann, Christian Spurk, Jörg Steffen, Stephan Busemann, Lucia Ingala-Hornung, Stefania Racciopa, Ulrich Schäfer, Maite Melero, Pavel Pecina, Chris Callison-Burch, Matt Post, Barry Haddow, Hieu Hoang, Juri Ganitkevich, Adam Lopez, Jan Hajic, Ondrej

Bojar, Manfred Pinkal, Stefan Thater, Magdalena Wolska, Ursula Kröner, Annemarie Friedrich, Sabine Hunsicker, Silke Theison, Philipp Koehn, Alon Lavie, Marcello Federico, Nick Ruiz, James Hodson, Alex Fraser, David Lemm, Aljoscha Burchardt, Georg Rehm, Peter Adolphs, Eleftherios Avramidis, Li Hong, Daniel Grasmick, and Martin Kay. I might have forgotten a name here and there, but you know who you are.

Last but not least, I want to thank all my friends who have not seen too much of me, especially during the final phase of my thesis work:

"It is finally done!"

Cheers and best,

Christian

Vancouver, November 2013.

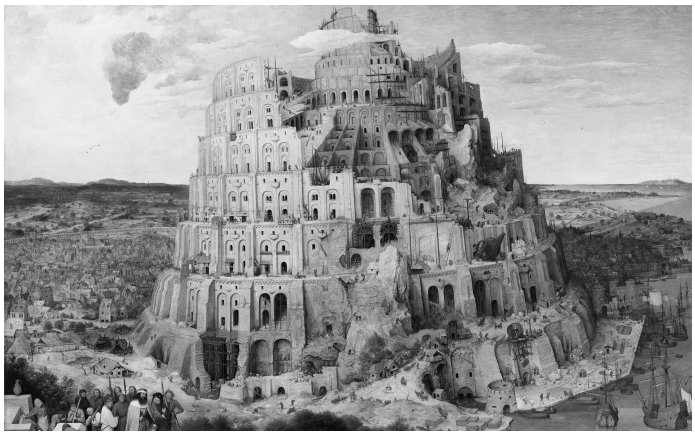
Table of Contents

List of Figures

List of Tables

List of Algorithms

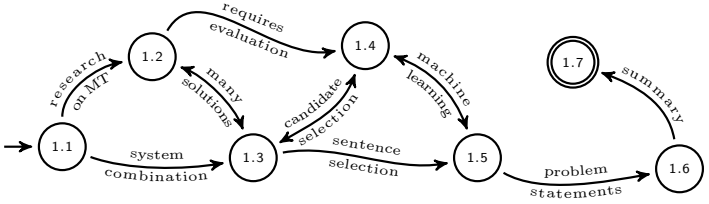
Preface



– Pieter Bruegel the Elder: The Tower of Babel, circa 1563

"Now the whole earth had one language and the same words. And as people migrated from the east, they found a plain in the land of Shinar and settled there. And they said to one another, "Come, let us make bricks, and burn them thoroughly." And they had brick for stone, and bitumen for mortar. Then they said, "Come, let us build ourselves a city and a tower with its top in the heavens, and let us make a name for ourselves, lest we be dispersed over the face of the whole earth." And the Lord came down to see the city and the tower, which the children of man had built. And the Lord said, "Behold, they are one people, and they have all one language, and this is only the beginning of what they will do. And nothing that they propose to do will now be impossible for them. Come, let us go down and there confuse their language, so that they may not understand one another's speech." So the Lord dispersed them from there over the face of all the earth, and they left off building the city. Therefore its name was called Babel, because there the Lord confused the language of all the earth. And from there the Lord dispersed them over the face of all the earth."

– Genesis 11:1-9



1 Introduction

Language "makes infinite use of finite means".

– **Wilhelm von Humboldt**: *Über die Verschiedenheit des menschlichen Sprachbaues und ihren Einflußauf die geistige Entwickelung des Menschengeschlechts* [von Humboldt, 1836] as quoted in [Chomsky, 1965]

"The limits of my language mean the limits of my world."

– **Ludwig Wittgenstein**: *Tractatus Logico-Philosophicus, Proposition 5.6* [Wittgenstein, 1922]

"Language is the source of misunderstandings."

– **Antoine de Saint-Exupéry**: *Le Petit Prince, Chapter XXI* [de Saint-Exupéry, 1943]

1.1 Translation Enables Understanding

Machine translation (MT) is a complex yet fascinating problem. Contrary to automatic speech recognition (ASR) which is only dependent on fragments of recorded speech in some language L that have to be mapped into text chunks in the *same* language, MT approaches also have to somehow model and capture the implicit transfer of words, phrases and concepts into another, *different* target language L_2. The latter problem is more complicated than the former. Due to the generative nature of language, as alluded to by Wilhelm von Humboldt already in 1836 [von Humboldt, 1836], there exists a plethora of potential, valid and correct translations for any given sentence.

Chomsky adopts this idea of a *"creative" aspect of language use* in his seminal work on the theory of syntax [Chomsky, 1965, p. *v*]. Given this property of language, it becomes evident that machine translation of text necessarily is a hard problem. A finding that is neither recent nor new. Instead it seems that philosophers and linguists alike identified the complexity and difficulties of natural language processing already long ago and that these still apply.

Adopting Wittgenstein's argumentation [Wittgenstein, 1922, § 5.6] it is clear that language is an integral part of human life. This claim holds both on the level of individual human beings but also for larger, social groups. Language can enable and support communication, hence fostering collaboration and progress, or represent an obstacle that hinders the free flow of information and eventually leads to conflict. Without a common language there is only little communication possible, or none at all. Without adequate communication it is difficult to form groups or work collaboratively. Therefore, language and linguistic competences do matter. Wittgenstein himself refers to the concept of *language as a limiting factor*, which can be interpreted as language being a fundamental factor that enlarges—or constraints—the world of the human being. While language is certainly not the only factor which influences exchange between human beings, it seems reasonable to state that an improved understanding among peoples could reduce the number of crises, wars and economic imbalances. This is what de Saint-Exupéry captures in his quote from [de Saint-Exupéry, 1943, Chapter XXI]: misunderstanding is caused by wrong or missing use of language. Translation as a linguistic process is a tool that empowers humankind to avoid such problems and helps to develop a greater and better understanding.

Following the technological advancements of the 20th Century, the amount of *language data* such as newspaper articles, television or radio broadcasts or

digital content available from the internet is steadily and quickly increasing. In its print edition from February 25, 2010, the British newspaper *The Economist* featured an article titled "Data, data everywhere"[1] in which the explosion of digital data is discussed and the implications with regards to storing all those *big data* are explained. In summary, the amounts of information available surpass existing storage capacities, leading to information loss considering the estimated growth rate of digital data. *"Information has gone from scarce to superabundant"* which means that the language technology sector is faced with challenges concerning what data to use and what not. Also the question of *data persistency* needs to be addressed to avoid losing important pieces of information. As a successor of the industrial society we are now living in what is called the *information society*. Knowledge and data are expressed using natural languages which have to be translated to enable their usage in different language communities. As the world is faced with a trend towards globalisation in many areas of human everyday life, translation has a central role in this process. And given the realities of digital content production, translation technology is challenged by massive amounts of such data.

Translation also constitutes a huge (and ever-growing) economic market. Estimates by the European Commission indicate that *"since 2008 the demand for the language and translation business in the European Union has been on the rise, growing from 8.4 billion Euro to about 10 billion Euro today"*.[2] In 2011 alone, *"more than two million pages have been translated for the European Commission"*. To achieve this throughput *"the EU itself employs as many as 5,300*

[1] Source: `http://www.economist.com/node/15557443`, retrieved November 24, 2012.

[2] Quotes originate from Ms. Contino, Head of Unit of Multilingualism and Translation Studies at the Directorate General for Translation. Taken from a presentation she gave at DCU on September 13, 2012. See `http://dculs.dcu.ie/dcu-language-services-news/the-translation-industry-and-the-european-union/` for the corresponding news item.

translators and interpreters – 2,500 of which are working in the Directorate General for Translation (DGT)" with estimated yearly costs of around 300 million Euro.[3] In its ISA programme (Interoperability Solutions for European Public Administrations), the European Commission reserves funding for the design, development and deployment of a "Machine Translation Service by the EC" (MT@EC).[4] According to Andreas Eisele, project manager MT at DGT, *"a real-life trial of the MT@EC service has been used by more than 800 users inside DGT between May 2011 and November 2012 to translate more than four million pages."*. As part of the MT@EC program, the European Commission plans the release of more data such as translation memories from the Official Journal of the EU (DGT-TM) in addition to the already released Acquis Communautaire (DGT-Acquis) [Steinberger et al., 2006].[5]

As the European Union aims to *"protect linguistic diversity and to promote knowledge of languages"*[6] it seems clear that the amount of translation work the DGT is handling continues to grow, similar to the *explosion of digital data* mentioned above. Faced with this challenge, the Directorate General for Translation explores how machine translation methods can be utilised to cope with the high volume of translation requests, *complementing* the work of human translators and interpreters. Current state-of-the-art MT tools may not be usable for tasks requiring high quality translation output but they may well be ready for some of the translation tasks at hand. Given the sheer amounts of data to be translated the assumption that machine translation tools would replace human translation professionals has no sound basis. This especially holds when considering the exponential growth of data compared to the number of trained translators. Translation technology thus should not

[3] http://ec.europa.eu/dgs/translation/faq/index_en.htm

[4] http://ec.europa.eu/isa/documents/isa_wp_second_revision_2012_annex_en.pdf

[5] http://www.w3.org/International/multilingualweb/luxembourg/slides/31-pilos.pdf

[6] http://ec.europa.eu/languages/orphans/faq_en.htm#9

be perceived as threat but rather as chance. This is nicely stated in a quote from [Kelly and Zetzsche, 2012]:

> "Translation software is not making translators obsolete.
> Has medical diagnostic software made doctors obsolete?"
> – **Nataly Kelly, Jost Zetzsche**: *Found in Translation: How Language Shapes Our Lives and Transforms the World* [Kelly and Zetzsche, 2012]

Unsurprisingly, research efforts on machine translation technology have seen a boost over the last decades, e.g., in the EuroMatrix (IST-034291) and EuroMatrixPlus (ICT-231720) projects or the Network of Excellence T4ME (FP7-249119)[7]. Similar to developments in Europe, research funding for MT projects has also been substantially increased in the USA and Asia. The same holds true for research groups and laboratories of companies such as Microsoft, Google or IBM. As a result research on MT has achieved a lot of progress over the last two decades. Since the 1990s statistical approaches have largely dominated MT research activities. The increasing availability of *parallel or pseudo-parallel data* has enabled data-driven methods that have evolved into sophisticated and robust translation tools which exhibit strong performance in terms of translation quality. A comprehensive summary of the early history of machine translation can be found in [Hutchins, 1986]. An overview on recent developments in MT research is provided in [Koehn, 2010, Chapter 1.2]. Section 1.2 gives a quick summary, highlighting several aspects relevant to this thesis.

In parallel, public awareness related to MT issues has greatly increased. The EuroMatrix project started a series of so-called "Machine Translation Marathons" (MTM) in which interested persons could learn about translation methods and models. Next to its summer/winter school aspect, the MTM

[7] More information available on individual project websites `http://www.euromatrix.net/`, `http://www.euromatrixplus.net/`, and `http://www.meta-net.eu/projects/t4me/`

events also host research talks and invite researchers and students to work on actual projects. Last but not least there is an open-source convention in which tools for MT training, translation or evaluation are presented and discussed. Similar to the continued success of the Linux operating system[8] and welcomed by public funding agencies the amount of source code and data which is released under permissive, open licensing terms has broadened widely in recent years. In his book "Just for Fun: The Story of an Accidental Revolutionary", Linux creator Linus Torvalds explains the reasoning behind the open source idea:

> "The theory behind open source is simple. (...) Anyone can improve it, change it, exploit it. (...) Think Zen. The project belongs to no one and to everyone. When a project is opened up, there is rapid and continual development."
>
> **– Linus Torvalds, David Diamond**: *Just for Fun: The Story of an Accidental Revolutionary* [Torvalds and Diamond, 2001]

As many of these open-source tools can also be used commercially it is fair to state that research endeavours had a positive impact on the translation market, empowering small and medium-sized businesses. Notable systems that have originated from research funding are:

- the Moses decoder as described in [Koehn et al., 2007];
- the Joshua toolkit, see [Li et al., 2009];
- the cdec framework explained by [Dyer et al., 2010];
- the Jane decoder published as [Vilar et al., 2010].

Next to software packages there also exists a multitude of *multi-lingual data* such as parallel corpora, translation memories or terminology databases. We

[8] http://www.kernel.org/

list a selection of freely available parallel corpora that are widely used in machine translation research below:

- the Canadian Hansards Corpus;[9]
- the Europarl corpus provided the Parliament of the European Union and processed as parallel texts suited for MT research [Koehn, 2005];
- the Acquis Communautaire distributed by the JRC of the European Commission [Steinberger et al., 2006];
- the UN Corpus harvested from United Nations websites as part of the EuroMatrixPlus project [Eisele and Chen, 2010].

The Linguistic Data Consortium (LDC)[10] also collects and distributes a lot of useful resources. The Evaluations and Language Resources Distribution Agency (ELDA)[11] performs similar functions in Europe. Access to data is an important prerequisite for machine translation systems. The emergence of agencies such as the LDC or ELDA and the increasing availability of *source data*, thus, has had a catalytic effect on the field of language technology.

After having set the context of this research undertaking we continue with an overview on machine translation research in Section 1.2. We briefly describe a selection of different scientific approaches and research paradigms. A central finding from this analysis is that different translation techniques have their individual strengths and weaknesses. Often, these qualitative variations are in fact *complementary* which, provided our assumption holds, means that a combination of translation output may yield an improved overall quality of the resulting combined or *hybrid* translation output. Seminal work by [Frederking and Nirenburg, 1994] conducted as part of the German Verbmobil project (1993–2000) adopts this conceptual idea and reports on

[9] http://www.isi.edu/natural-language/download/hansard/

[10] http://www.ldc.upenn.edu/

[11] http://www.elda.org/

improvements in terms of translation quality for the combination of several source systems.

Following this section, we discuss system combination approaches and describe the current state of the art in Section 1.3. Several techniques are presented and discussed. Parallel to the increasing importance and performance of statistical methods, a common approach in system combination research is *confusion network* decoding. Here, candidate translations are chunked into smaller units and aligned into a connected graph. Using probability scores obtained from the individual systems (or computed otherwise) this graph can then be traversed to find the optimal *consensus translation*. An obvious limitation of this method is that it may introduce errors, both syntactic and semantic, due to the implicit partitioning of the candidate sentences into smaller chunks. *Sentence selection* mechanisms aim to overcome this problem by avoiding modifications of the given candidate translations. At the same time, it is clear that these methods suffer from their disability to integrate knowledge on the sub-sentential level. To remedy these issues, several hybrid MT approaches have been proposed that aim to integrate multiple sources of knowledge into a translation engine, either on a shallow linguistic level or embedded deeply into the decoder.

The evaluation of machine translation quality is an important task in MT research. We introduce this topic in Section 1.4. Statistical systems rely on evaluation scores during tuning. MT systems from all underlying technological paradigms are developed with increasing translation quality as main goal and thus require techniques for quality assessment. We present several approaches that allow to perform this assessment. These range from manual annotations, e.g., on the translation output's fluency or its perceived adequacy to fully automatic metrics such as BLEU [Papineni et al., 2002] or

Meteor [Denkowski and Lavie, 2011]. We also cover various error distances such as TER [Snover et al., 2006] or its successor TER*p*lus [Snover et al., 2009]. As automatic methods require the availability of one or several reference translations, they cannot always be applied. Furthermore, their correlation to human judgment is not unanimously agreed on. Recently, the evaluation of translation output without access to reference data has become an area of active research by itself. We illustrate how these *quality estimation* techniques work. Our overview on MT evaluation concludes with a brief discussion of various tools for quality assessment, translation ranking, and post-editing.

Before turning to the formulation of problem statements, we provide an overview on machine learning (ML) techniques in Section 1.5. We discuss their application in machine translation or quality estimation, paying special attention to potential usefulness for hybrid approaches. Machine learning by itself is a well established research problem. Research activities have resulted in a large number of algorithms such as the Perceptron [Rosenblatt, 1958] or Winnow [Littlestone, 1988] algorithms that learn incrementally from labeled training data, or more complicated kernel-based methods such as support vector machines (SVM) as described by [Vapnik, 1995] or the more recent relevance vector machine (RVM) by [Tipping, 2001]. Binary classification problems usually can be solved achieving a high performance with respect to prediction accuracy on unseen input data. In fact, n-ary classification is often modelled by decomposition into pairwise—thus binary—classification problems. Machine learning can also be applied for system combination or quality estimation. Feature vectors are computed on the level of individual systems which are then used as input for one or several classification models. *Interpretation* of classification results is performed in some post-hoc process, depending on the actual task under investigation. By contrast, we pursue a

different strategy and re-formulate the feature vector definition to explicitly model comparison between two given systems A, B. The paradigmatic shift from single to what we call *joint feature vectors* is a central contribution of this thesis work. As we will see later, the application of joint feature vectors can help to perform sentence selection with high accuracy. Furthermore, it allows us to adapt sentence selection to specific characteristics of given system pairs.

We conclude this introductory chapter by formulating the set of research hypotheses and the corresponding problem statements that will be examined in the remainder of this thesis work. Afterwards, we briefly summarise the contents of this chapter and provide an outlook on the upcoming chapters.

1.2 Machine Translation Methods

Machine translation is one of the oldest research problems in the fields of computer science and artificial intelligence. Initially, it was thought an easy task, a perception that changed over time as it turned out that the underlying computational problems are indeed very complex. The decoding process of a statistical MT decoder needs to find a solution to the following equation.

Definition 1. *Let e denote one translation of some given, foreign sentence f under the current translation model. The statistical translation problem is defined as finding the best*—most probable—*translation $\hat{e}$ for the observed foreign sentence f. Formally, this can be expressed as follows:*

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(e \mid f)$$

The corresponding search problem can in fact be reformulated into an instance of the infamous *Traveling Salesman Problem* which is known to be in the class of so-called *NP-complete* decision problems. This has been shown by

[Zaslavskiy et al., 2009]. Simply put, this means that there exists no known method that guarantees to compute all solutions to the given problem in a reasonable amount of time. It is, however, possible to approach the solution using approximation algorithms. Of course, this also means that there are no guarantees towards the optimality of the achieved results.

An Overview on MT History

After the end of the Second World War in 1945 and with the beginning of the Cold War between the United States of America and the Soviet Union, a new demand for translation of Russian texts arose. The parallel advent of computing machinery led to the belief that machine translation would soon be able to take over translation duties from human interpreters. The *Georgetown-IBM experiment* which took place on January 7, 1954, fuelled such beliefs by successfully translating sixty Russian sentences into English in what was called "*a Kitty Hawk of electronic translation*". It was assumed that in "*five, perhaps three years hence, interlingual meaning conversion by electronic process (...) may well be an accomplished fact.*".[12] More details on this seminal event in the history of MT research can be found in [Hutchins, 2004].

The first machine translation systems focused on translation by *direct transfer*. The following years showed an increase in both research funding and activities in the emerging field of computational linguistics. This changed with the release of the famous ALPAC report [Pierce et al., 1966] in 1966. The authors expressed doubt that machine translation would actually be more cost effective than human translation in the near future and without advances in our knowledge about processing human language. As a result,

[12] Source: `http://www-03.ibm.com/ibm/history/exhibits/701/701_translator.html`, retrieved on January 28, 2013.

funding was drastically cut and the first wave of MT research came to an abrupt end.

Interest renewed in the 1970s when the first *rule-based* MT systems (RBMT) arrived. Such approaches aim to model linguistic phenomena for both source and target language, following some grammar formalism or linguistic theory. Translation is implemented by a *linguistic transfer* step that maps linguistic structures obtained during the *analysis* of the source sentence to *equivalent* structures in the target language. A final *generation* phase ensures proper linguistic treatment w.r.t. the specific target language, fixing, e.g., agreement or word order. Important milestones include the SYSTRAN translation engine and the Eurotra project, funded by the European Commission. The latter in hindsight turned out to be a failure as "*fully automatic high quality translation*" could not be achieved. This dampened optimism and subsequently complicated the acquisition of European research funding for MT research.

As the construction of RBMT systems is a lengthy and expensive process, research on automatic, statistical learning of translation equivalences started. The underlying idea is that, given a large quantity of English text plus the corresponding translations into a foreign language, one can statistically infer translation probabilities for words or larger chunks of text. The fundamental mathematics for statistical machine translation were developed in the early 1990s at IBM's Thomas J. Watson Research Center [Brown et al., 1993]. The concept of statistical machine translation, interestingly, had already been formulated by Warren Weaver in his famous "Translation" memorandum, written in July 1949[13]. It was later reprinted and published by William Locke in [Locke and Booth, 1955]. Statistical methods have since become the prevalent technological paradigm and primary object of research in the area

[13]It is also available at `http://www.mt-archive.info/Weaver-1949.pdf`

of machine translation.

Recent Research Activities

The 2000s brought an increasing number of statistical MT resources which could freely be used and, sometimes, even be modified due to openly available source code. As a side effect, the number of researchers working on machine translation greatly increased over the last decade. The availability of shared resources and access to the same tools allowed for a better comparison of results and helped to attract new researchers to the field. Notable additions to the state of the art have been achieved in the following areas:

- continuous space language models [Schwenk et al., 2006];
- factored translation models [Koehn and Hoang, 2007];
- hierarchical translation models [Chiang, 2007];
- paraphrasing for translation [Callison-Burch, 2007];
- improved language modelling [Stolcke, 2002, Federico et al., 2008];
- stream-based models [Levenberg et al., 2010, Levenberg et al., 2011];
- open-source RBMT research [Forcada et al., 2011].

At the same time, a trends towards *harmonisation* across different decoder implementations can be observed. This, again, improves comparability of results and supports collaborative efforts. Machine translation research has also lead to commercialisation activities in response to the growing demands of the global translation/localisation markets. An important trend over the last couple of years lies in a focus shift towards *hybrid MT* systems. These systems aim at the combination of resources and techniques from various technological backgrounds, e.g., rule-based and statistical approaches. In this thesis, we investigate how hybrid machine translation can be modelled and implemented using machine learning tools.

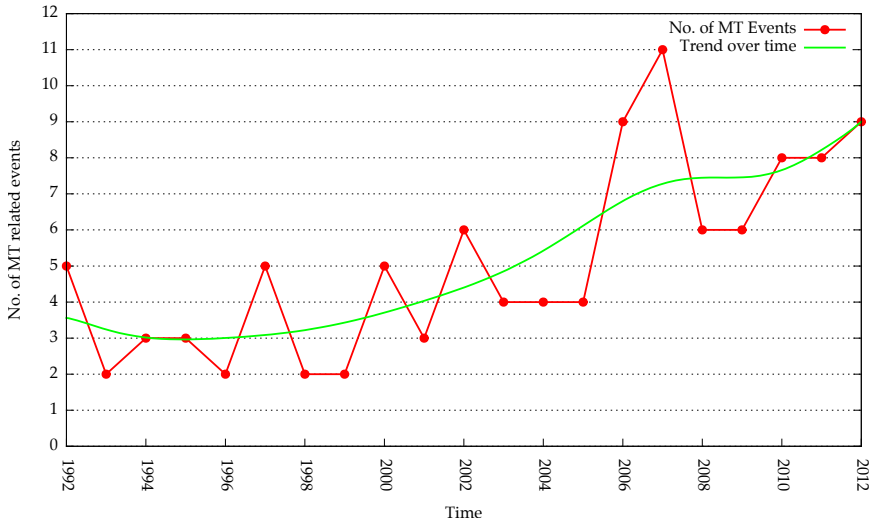


Figure 1.1: Number of MT related events/corresponding trend over time

A Research Field with Growing Impact

Next to research efforts, several workshops and conferences such as, e.g., the "Workshop on Statistical Machine Translation" (WMT) have been established, further attracting interest in the research topic. The number of published papers on translation techniques has seen a rapid growth since the early 2000s, a manifestation of the field's growing importance. Figure 1.1 gives an overview on the number of MT related events from 1992 until 2012 as they are listed in the MT Archive. The trend shows how machine translation research activities—and, thus, interest in MT—increased over the years.

Machine translation has also become part of everyday life and is used both for business and leisure. Free services such as Google Translate[14] or Bing Translator[15] have changed the way users interact with foreign language content on the internet. The general perception by lay users is that machine translation should nowadays be able to produce high quality output. Users expect quick turnaround times and flawless output, turning the deployment

[14] http://translate.google.com/
[15] http://www.bing.com/translator/

and maintenance of high quality translation services into an even harder task. MT research, hence, is of high importance—both scientifically and sociologically.

The wealth of machine translation methods and the high level of research activity promise the discovery of more efficient approaches which are, in the long run, able to generate such high quality translation output. One area of research lies in the investigation of so-called *system combination* methods. These consider translation output from several machine translation systems (or several translation hypotheses from a single system) and try to generate a joint translation with an improved, overall translation quality. The rationale is that a clever combination of translation fragments should result in better translation output. A schematic overview on the basic problem setting for system combination approaches is depicted in Figure 1.2. Such systems are, in a sense, hybrid MT systems.[16] We will cover combination methods in more detail in the next section.

1.3 System Combination Approaches

As we have seen in the previous section, machine translation is possible using various methodological paradigms, each having its individual strengths and weaknesses. Considering the wide range of available methods it makes sense to think about *combining translation output*, which is also called *multi-engine machine translation* (MEMT) or *hybrid MT*. Think, for instance, about lexical coverage in a machine translation model. For statistical systems, it only depends on the contents of the parallel training data. "*The more data, the bet-*

[16] Typically, a system is considered hybrid if it combines translation output which has been generated by MT systems implementing different technological paradigms, e.g., statistical and rule-based MT. We use the term hybrid in a broader sense and include system combination approaches which work on only one methodological paradigm.

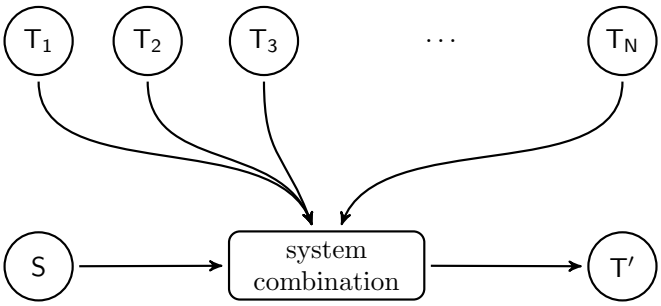


Figure 1.2: Overview on the basic problem setting for system combination

	Linguistic phenomena				
Paradigm	Syntax, Morphology	Structural Semantics	Lexical Semantics	Lexical Adaptivity	Lexical Reliability
RBMT	++	+	−	−−	+
SMT	−	−−	+	+	−

Table 1.1: Informal comparison of RBMT and SMT methodologies w.r.t. their strengths and weaknesses. Adapted from a EuroMatrix Plus presentation.

ter" is a phrase often heard in this respect. The situation is different for rule-based systems. As RBMT engines rely on linguistically informed rules and knowledge bases whose production is both expensive and time-consuming, their lexical coverage cannot adapt to new trends as quickly as SMT.

Complementary Strengths and Weaknesses

Quite often errors are complementary among the different MT methods. Thus, in theory, a clever combination of multiple candidate translations *could* yield a translation with improved, overall translation quality. Table 1.1 gives a quick overview on how 1) rule-based and 2) statistical machine translation methods handle a selection of linguistic phenomena. Note how strengths and weaknesses are indeed complementary. The table has been adapted from FP7 funded research project EuroMatrix Plus (ICT 231720) which conducted

Algorithm 1 Decision problem for system combination approaches

Require: set of source sentences S,
Require: translation output from N systems, $T = \{T_1, T_2, \ldots, T_N\}$.
Ensure: $|S| = |T_1| = |T_2| = \ldots = |T_N|$

1: $T' \leftarrow$ **compute_best_translation**(S, T) ▷ Compute best translation given input data
2: **return** T' ▷ Return combined translation output

research on hybrid machine translation. More information on activities in EM+ WP2 can be found in the corresponding project reports and publications, e.g., [Federmann et al., 2011] or [Wolf et al., 2011]. One of the lessons learnt during our work on the combination of RBMT and SMT technology was that integration on a deeper linguistic level is a complex task. The addition of parallel data extracted from a given SMT system proved to be problematic as these phrases contained too little linguistic annotation to be included in the lexical resources of the rule-based engine. This also affected the amount of data which could be integrated into the rule-based MT system—as we had to augment phrase pairs with additional, linguistic annotation—which in turn meant losing one of the advantages of statistical learning, namely the power of inference from very large data sets. While the research in the EuroMatrix Plus project was successful and resulted in several interesting extensions of the given rule-based system, there remains a lot of potential for future improvements.

Combination Approaches

There exists a wide range of system combination approaches. Many of these aim at solving the decision problem shown in Algorithm 1. Other approaches may work sequentially or follow some other methodology. In this thesis, we focus on parallel combination by sentence selection.

Google		Understanding		a language is an old		dream			.
Bing		Understanding		a language is an old		dream	of	humanity	.
Systran		Understanding		a language is an old		dream	of	mankind	.
Lucy	The	understanding	of	a language is an old	mankind	dream			.

They differ in design and implementation of **compute_best_translation** which computes the final combination result. A common solution to approach this problem is the application of *confusion network decoding*. Using word alignment techniques, individual candidate translations T_i are aligned to a pre-selected, designated translation "*backbone*" or "*skeleton*". The candidate translations form a network, i.e., a connected graph. Edges between different target words are labeled with transition probabilities—in this case, translation probabilities or estimated future costs obtained from the decoding engine—thus spanning the network of all possible generations considering the alignment and vocabulary from the given set of candidate translations. An example of a confusion network is shown in Figure 1.3. Note that the possibility of "*empty*" or so-called ϵ transitions allows for the generation of a large number of translations which were not originally contained in the set of candidates. A lot of these, however, do not represent *valid* sentences due to combinatorial effects such as, e.g., double prepositions, wrong agreement or other phenomena. On the other hand, confusion network decoding is able to generate translations which contain good parts from several candidate translations, provided their transition probabilities promote the corresponding decoding paths throughout the network.

We will provide a more detailed discussion of this technique as part of our literature review in Chapter 2. As a side note, it is also possible to apply confusion network decoding to the task of *n-best list re-ranking* for a single machine translation system. System combination using confusion networks has become the dominant methodological approach in recent years,

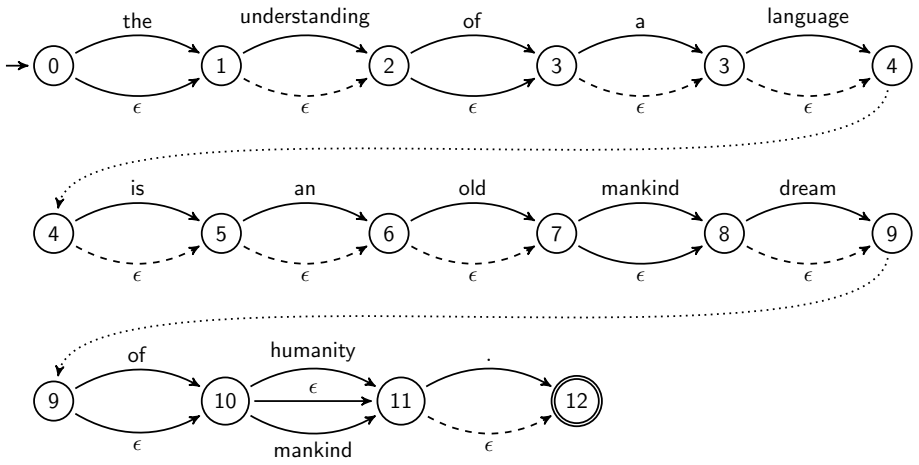


Figure 1.3: Example of a confusion network graph encoding four different translations of German sentence "*Das Verstehen einer Sprache ist ein alter Menschheitstraum*". Note how many invalid sentences can be produced. Dashed arrows illustrate the effect of additional, general ϵ transitions between nodes which would contribute further derivations, many of which invalid.

e.g., in the system combination tasks undertaken as part of the Workshop on Statistical Machine Translation (WMT '09–'11).

Sentence Selection

As we have remarked in the previous section, transitions in a confusion network may lead to ungrammatical and erroneous translations. This even holds if all given source sentences were perfect translations, due to the "generative" nature of the decoding process. It is also clear that any errors introduced in the word alignment phase can proliferate and may cause degradations of the resulting translation. The selection of the underlying translation backbone has an influence on the outcome as well.

Considering these shortcomings there has also been (somewhat scarce) research on the problem of *sentence selection*. Given a set of candidate translations, the combined translation is computed by selecting the best among the given candidates, in unaltered form: *e pluribus unum, immutatum*. It

is obvious that such an approach cannot "fuse" phrasal phenomena from multiple sentences into the final translation. On the other hand, it also is impossible that the chosen translation is altered—i.e., potentially degraded in terms of translation quality—in any way.

Selection mechanisms have been studied by [Hildebrand and Vogel, 2008, Hildebrand and Vogel, 2009, Hildebrand and Vogel, 2010]. Overall, interest in this research topic has, however, been limited; most likely due to the prevalence of aforementioned confusion-network-based methods. Improved techniques able to solve the selection problem on the sentence level could also be applied on the level of sub phrases, making them an interesting area for further research in our view.

1.4 Evaluation of Translation Quality

Machine translation research and MT system development rely on *evaluation methods* that measure and compare translation quality. Such assessments can be performed by human annotators who score one or several translations w.r.t. to 1) fluency, i.e., answering the question "*How likely is this translation to be a well-formed sentence in language* L_X?" and 2) adequacy, i.e., answering the question "*How well does this translation convey the intended meaning of the source sentence?*". Usually, both questions are scored using a 1–5 range, where 1 means "*Not at all*" and 5 denotes "*Perfect*". While this methodology allows for a fine-grained inspection of translation output, the exact interpretation of scores and the subtle differences between individual scoring categories make it a time-consuming effort with surprisingly low inter-annotator agreement.[17] Even worse, the notions of both fluency and

[17] See, e.g., the results paper [Callison-Burch et al., 2007] from WMT '07 where only *fair agreement*, i.e. $0.21 \leq \kappa \leq 0.4$, among annotators could be observed for both fluency and

adequacy are hard to "grasp" and model with automatic means.

As the training and tuning processes of SMT systems require evaluation metrics which can be computed (relatively) quickly, research turned towards *n-gram overlap scores* such as BLEU [Papineni et al., 2002]. These take the translation output from some system and compare it to one or several given reference translations by measuring overlap on the level of n-grams. While such scores are easy to compute and hence attractive from the viewpoint of algorithmic complexity, their correlation with human annotation results still remains a topic of active research.[18] Notable efforts have been undertaken as part of the yearly WMT workshops, with special "*shared evaluation tasks*" being held from 2008 until 2010. Irrespective of improvements in the field of automated metrics and despite its known shortcomings, BLEU remains the most frequently used metric for automatic evaluation to date.

In our experience, the Meteor score [Denkowski and Lavie, 2011] proves more useful in practice. It applies advanced techniques such as, e.g., word synonymy lookup using WordNet [Fellbaum, 1998][19] and has also been built in a way that makes sentence-level scores usable. For this reason, it will be used prominently in the remainder of this thesis. There also exist several *translation edit/error rates* such as *TER*, *PER*, *WER* which implement versions of the *Levenshtein distance* on the word level.

A main issue with all reference-based evaluation metrics is the *creative aspect of language*: given a single source sentence the set of valid translations can become very large. It seems infeasible to pre-compute or store all possible translations. And even if this was possible, there would be no automatic

adequacy evaluation as well as sentence ranking.

[18] Some metrics, such as the *de-facto standard* BLEU, do not even generate scores which are proven to have a high correlation on the sentence level which makes it very hard to use these scores for error analysis on the sub-corpus level.

[19] Of course, this can only be applied for languages where such resources are available.

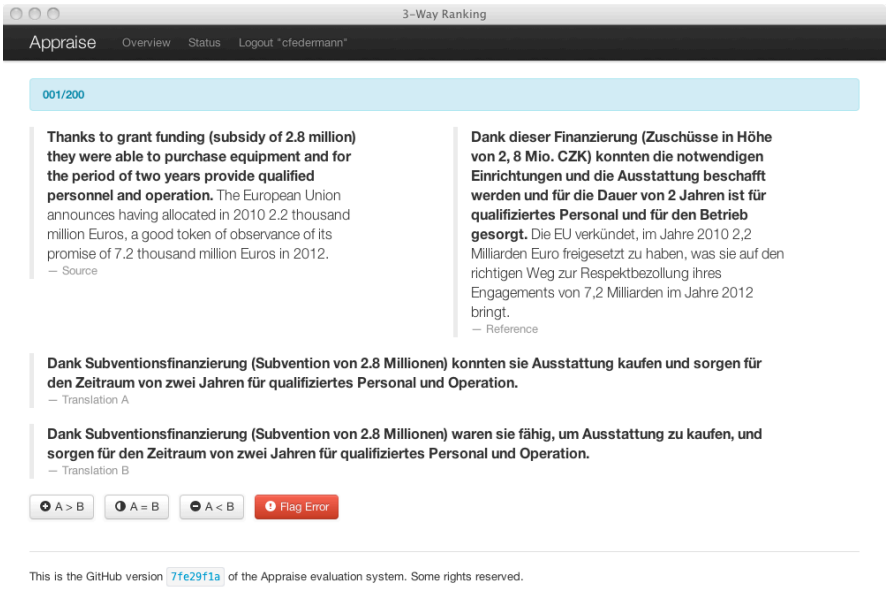


Figure 1.4: Screenshot of "3-Way Ranking" as implemented by Appraise

way of differentiating between those "possible" translations in terms of translation quality. Recently, research on generating large *reference networks* has been started. See [Dreyer and Marcu, 2012] for more related information. As the construction of such networks, again, is a very time-consuming task, it cannot readily be applied to SMT tuning or evaluation.

Human annotation of machine translation output has been extended to additional evaluation tasks, which are expected to be easier to perform and to result in higher levels of inter-annotator agreement. Examples for such annotation tasks are:

- ranking comparison of several translation candidates;
- binary ranking comparison;
- sub-phrase comparison between two translation candidates;
- error classification of one candidate translation;
- (minimal) post-editing of one given translation;
- gisting classification for one translation candidate.

As part of the research conducted for this thesis, we have developed a tool for MT evaluation named *Appraise* [Federmann, 2012b] which implements the aforementioned tasks. The tool has been released under a permissive open-source licence and is available online from the author's GitHub repository.[20] A screenshot of the "binary ranking comparison" interface is depicted in Figure 1.4. Several other tools have been produced and made available as well, e.g., PET [Aziz et al., 2012]. The main issues with human annotation of translation output still remain valid: 1) level of inter-annotator agreement and 2) overall annotation speed. Due to the increased availability of improved user interfaces and specialised tools, these issues will become less relevant over time, eventually leading to better and more helpful evaluation of machine translation quality.

1.5 Methods Using Machine Learning

As we have seen, evaluation of machine translation output is a hard task. This holds both for human annotations which are time-consuming and thus expensive and for automatic scoring metrics whose correlation with human judgment is not sufficiently clear yet. Evaluation is difficult because the differences between a given translation and some reference are hard to be *intuitively grasped* by human annotators and challenging to model for automatic processing. As the combination of several translation candidates is based on the comparison of the individual sentences, we are faced with a similarly complex problem.

It is clear that any automatic solution requires careful modelling of the underlying decision problem to stand a chance of successfully generating improved translation output. Due to the differences between the various

[20] See `https://github.com/cfedermann/Appraise`

machine translation paradigms, research efforts on the application of suitable machine learning tools have been intensified. Such ML techniques can potentially better handle the diverse feature sets produced by the given MT systems, especially when statistical decoders with large amounts of features and factors are considered.

Example: Hybrid MT using NP substitution

We give a brief example to further illustrate the aforementioned complexity of multi-paradigm system combination: As part of our research within the EuroMatrix Plus project, we have worked on several hybrid MT architectures, aiming at targeted substitution of noun phrases within a translation template provided by a rule-based system. Initial research had shown that statistically learnt phrase tables often contain "*better*" (in the sense of more up-to-date and more fluent) translations for noun phrases than the RBMT engine under investigation. We implemented a hybrid system which aligned RBMT and SMT translation output and then replaced noun phrases within the rule-based template by their corresponding counterparts from the statistical system. See [Federmann et al., 2009].

The substitution process was controlled using a set of decision factors such as, e.g., *part-of-speech agreement* or *target language model scores* to avoid problems leading to degradations of the original translation template. The *decision flow* was designed manually after careful inspection of individual factors and their contribution towards a "good" or a "bad" translation. Later, it became clear that more factors would be required to improve the quality of the substitution process. The addition of such new factors, however, also required changes to the overall decision flow. These changes turned out to become more complex with every additional factor added. To cope with

such complexities, we applied machine learning tools such as *decision trees* to automatically infer the importance of individual factors (or *features* in machine learning terms). See [Federmann, 2012c, Hunsicker et al., 2012].

Machine Learning for Binary Decision Problems

The field of machine learning is investigated by a large and active research community. One of the primary ML problems is that of two-class, i.e., binary classification. There exist many competitive algorithms to solve this problem, often achieving high prediction accuracy. By re-formulating the decision problem that has been defined in Algorithm 1 from N-ary selection down to individual, pairwise comparisons between candidate systems, it would be possible to make use of state-of-the-art machine learning tools. Considering the fact that it also is more plausible to find features that allow to distinguish between two systems (rather than features that work for the complete set of N candidates) the application of binary classification for system combination seems a promising undertaking.

Quality Estimation

In fact, machine learning is already applied in the context of *quality estimation* (QE) methods. In contrast to evaluation techniques where reference text is available for comparison, QE approaches aim at predicting a ranking of given candidate translations without references available. Typical approaches use linguistic features such as, e.g., *parse probabilities*, *language model scores*, or *alignment links* for estimating the quality of the given translation. Note that this estimation process is performed on the level of individual candidate translations—the aggregation of individual classification results into the final ranking of candidates is computed in a subsequent, *post-hoc* computation

step. Effectively, this means that approaches following this methodology do not model an explicit comparison between two given systems. The 2012 edition of WMT featured a designated task on quality estimation methods.[21] The shared task has also been offered in 2013.[22]

ML4HMT Workshops

Work package 2 of the T4ME project has investigated how the combination of translation output obtained from several MT systems can be improved by the integration of machine learning methods. A series of workshops (ML4HMT '11/'12)[23] including an associated system combination task were organised. Participants of the shared task received a corpus containing translations from four MT engines, originating from rule-based and statistical backgrounds. Additionally, they received meta-data information extracted from each of the engines. As mentioned before, the set of meta-data, i.e., *potential features*, is heterogenous w.r.t. the types of individual features. This makes it complex to derive an understanding of their impact towards sentence selection. Thus, machine learning techniques were applied to solve the decision problem. Results from the shared tasks and descriptions of the participating systems are reported in [Federmann, 2011, van Genabith et al., 2012]. More details on the ML4HMT corpus are available in [Federmann et al., 2012].

In summary, machine learning approaches can be utilised to improve system combination efforts in the context of machine translation. The main problem on a methodological level is that features for candidate translations are evaluated in isolation only while final comparison results are produced in a post-hoc aggregation step. We will show in this thesis that it is beneficial

[21] See `http://statmt.org/wmt12/quality-estimation-task.html`
[22] See `http://statmt.org/wmt13/quality-estimation-task.html`
[23] See `http://www.dfki.de/ml4hmt/`

to explicitly model pairwise comparison of systems instead.

1.6 Problem Statements

In this introductory chapter we have explained why research on improved sentence selection algorithms for hybrid machine translation is a worthwhile undertaking. Even if such selection approaches are not able to combine sub phrases from several candidates into an improved translation, their ability to conserve characteristics generated by the source MT engine can be their competitive advantage. A powerful selection mechanism able to separate good translations from bad counterparts is required to implement such methods. In this thesis we will investigate several related research problems and discuss suitable solutions as well as observations from our experimental findings and their implications towards the state of the art. We also provide a brief overview on the contributions of this thesis and related publications.

Research Questions

This thesis aims at answering the following research questions:

- What level of translation quality can be achieved by using sentence selection approaches instead of confusion network decoding?
- What advantages do sentence selection algorithms have compared to n-gram combination approaches?
- What system combination quality is theoretically achievable?
- What is the benefit of applying machine learning using joint, comparison-based binarisation of features compared to single feature vectors?
- What algorithms can be applied for training and tuning a hybrid MT system based on pairwise, binary classification?

Thesis Contributions

The major contributions of this thesis are:

- We conduct the **largest meta-study on WMT results** published by the Workshop on Statistical Machine Translation (2007–2013);
- The definition of **joint, comparison-based binarisation of features** for machine learning of binary classification between several candidate translations;
- We show that **sentence selection approaches can perform** at the same level of translation quality as existing system combination methods;
- Our evaluation software for assessment of machine translation quality, **Appraise**, has become the official evaluation system of WMT 2013.

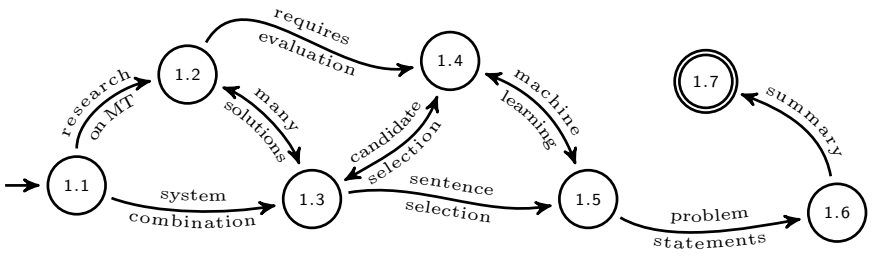
Related Publications

This thesis combines the following selection of peer-reviewed publications:

- Experiments on system combination and integration of rule-based and statistical methods into hybrid MT approaches have been submitted to the Workshops on Statistical Machine Translation 2007–2012;
- Our methodology—including the definition of joint, comparison-based binarisation of feature vectors—has been published in the proceedings of the 35th Annual German Conference on Artificial Intelligence in Saarbrücken, Germany [Federmann, 2012a] and in the proceedings of the Tenth Conference of the Association for Machine Translation in the Americas in San Diego, USA [Federmann, 2012d];
- The Appraise system for manual evaluation of MT output has been published in the proceedings of the Seventh International Conference

on Language Resources and Evaluation in [Federmann, 2010] and in the Prague Bulletin of Mathematical Linguistics [Federmann, 2012b];

– Appraise has become the official evaluation system of the Workshop on Statistical Machine Translation in WMT 2013 [Bojar et al., 2013];
– The ML4HMT corpus has been published in the Prague Bulletin of Mathematical Linguistics [Federmann et al., 2012]. Results from the corresponding shared tasks have been published as joint proceedings of LIHMT '11 and ML4HMT '11 in Barcelona, Spain [Federmann, 2011] and in the proceedings of the 24th International Conference on Computational Linguistics in Mumbai, India [van Genabith et al., 2012];

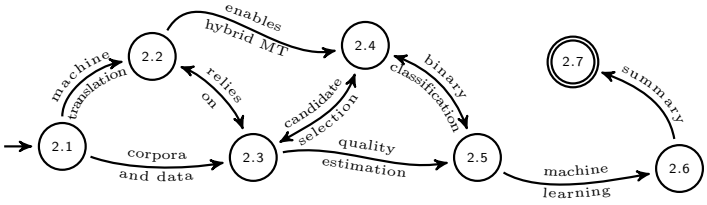


1.7 Chapter Summary

In this chapter we have motivated the problem of machine translation paying special attention to system combination approaches or hybrid methods. We have learnt that language technologies such as, e.g., MT have given rise to huge commercial markets and areas of high importance. Consequently, there is an active research community addressing algorithmic complexity and improving translation methods. Considering these widespread activities we have argued that system combination methods can lead to improved, overall translation output. We have presented confusion network decoding and sentence selection approaches. The latter have not yet been applied too often but they can be used in combination with machine learning methods with promising results. We briefly discussed MT evaluation techniques and ML algorithms before formulating the problem statements for this thesis and the contributions we have achieved to improve the state of the art.

The remainder of this thesis is structured as follows: Chapter 2 provides an overview on relevant background literature and introduces the current state of the research field. In Chapter 3, we investigate theoretical performance of sentence selection based approaches to system combination. We conduct the largest meta-study on results published from WMT 2007–2013. In Chapter 4, we introduce the notion of joint, comparison-based binarisa-

tion of feature vectors which represent one of the key contributions of this thesis. We compare them to single feature vectors, aiming to verify their superiority. In Chapter 5 we define a theoretical framework for system combination using machine learning and describe the experiments we have conducted to evaluate the quality of our combination approach. We provide comparison to results from state-of-the-art machine translation systems. We conclude in Chapter 6, highlight the contributions achieved during our thesis work and giving an outlook to future research questions that arise from our findings.



2 Literature

"Reading is to the mind what exercise is to the body."
– Richard Steele: in *The Tatler*, March 18, 1710.

"All men by nature desire knowledge."
– Aristotle: *Metaphysics*.

2.1 Introduction

We have already discussed many fundamental publications on which the research work described in this thesis is based on in Chapter 1. In this chapter, we provide brief summaries of selected publications in five topic areas: 1) machine translation, 2) corpora and data sets, 3) hybrid systems, 4) quality evaluation, and 5) machine learning. For each of these topic areas, we discuss key publications in the relevant literature in chronological order.

The remainder of this chapter is structured as follows: in Section 2.2 we describe research on MT methods. Second, we present relevant corpora and data sets (Section 2.3) and discuss hybrid systems (Section 2.4). Afterwards, we explain evaluation techniques in Section 2.5 before describing machine learning (Section 2.6). We conclude with a summary in Section 2.7.

2.2 Machine Translation

[**Brown et al., 1993**] Being one of the very foundations of statistical machine translation methodology, this paper describes what is nowadays called the **IBM models** for word alignment and translation between languages.

[**Chiang, 2007**] This paper introduces the notion of **hierarchical phrase-based translation** which has seen widespread use in recent years. The addition of sub phrases inside phrases allows for improved translation quality in terms of BLEU scores. Formally, the hierarchical model from this article represents a synchronous context-free grammar which is learnt from parallel corpora without syntactic annotations.

[**Koehn et al., 2007**] In this paper, the authors describe the **Moses** toolkit for statistical machine translation. Moses has attracted an active community of both researchers and software developers and is generally considered an open-source success story.

[**Li et al., 2009**] presents **Joshua** which is a Java-based toolkit for parsing-based machine translation. Similar to aforementioned Moses toolkit, Joshua has gathered a loyal following over time. Joshua first shifted focus on hierarchical and syntax-based translation models.

[**Dyer et al., 2010**] The authors describe the design and implementation of **cdec** which is a toolkit for statistical machine translation as well as other structured prediction models. By contrast to other frameworks, it was developed with machine learning in mind. Also, it is designed to scale from limited resources up to large cluster systems.

[**Forcada et al., 2011**] This paper discusses **Apertium** which is a free and open-source platform implementing rule-based machine translation. The system features a shallow-transfer MT engine and produces translations in a series of sequential processing steps.

2.3 Corpora and Data

[**Fellbaum, 1998**] **WordNet** is a large lexical database of English. It contains sets of cognitive synonyms, so-called synsets, for a large set of English words. Such information can be beneficial in the context of machine translation (to find additional, synonymous lexical phrase pairs) or as part of quality evaluation (it is, for instance, used by Meteor).

[**Koehn, 2005**] The **Europarl** corpus is a collection of proceedings from the European Parliament. It includes versions in 21 European languages and has become one of the most widely used training corpora for MT.

[**Steinberger et al., 2006**] This paper presents the **JRC-Acquis** corpus which is comprised of European Union documents in 22 official European languages. Data is taken mostly from the legal domain.

[**Eisele and Chen, 2010**] This paper presents the **MultiUN** corpus which is a multilingual corpus compiled from publicly available documents from the United Nations. It features Arabic, Chinese, and Russian as well as four other European languages.

[**Avramidis et al., 2012**] The authors describe the **ML4HMT** corpus. It is a richly annotated, multilingual parallel corpus for hybrid machine translation research. The corpus has been used as part of the ML4HMT workshop series in 2011 and 2012. It contains translations from MT systems implementing different methodological paradigms.

2.4 Hybrid Systems

[Frederking and Nirenburg, 1994] Seminal paper on **system combination**. The authors describe an implementation which is able to combine translation output, typically sub phrases, from three candidate systems.

[Hildebrand and Vogel, 2008] The authors present a system combination approach which is based on a **sentence selection** method. Considering n-best lists as input they compute the resulting combined translation by selecting one of the hypotheses available from the n-best list. This approach is theoretically comparable to our method.

[Federmann et al., 2009] In this paper, the authors describe a combination approach based on **factored word substitution**. One of the candidate translations is used as "translation template". Using a parser and word alignment information, the system can then substitute in noun phrases from additional candidate systems, generating a hybrid translation.

[Federmann, 2011] The author presents results from the **ML4HMT 2011** workshop and shared translation task. Results are reported in terms of automatic metric scores and an extensive, manual evaluation implemented using Appraise. An interesting finding from the results is that the Meteor evaluation metric is the only metric which correlates well with human judgments.

[Wolf et al., 2011] The paper describes a tool for **terminology extraction**, based on statistical word alignment. Using terminology lists obtained using this tool, the lexical database of an existing, rule-based MT engine is extended over time, resulting in a hybrid machine translation system.

[Federmann, 2012a] Being part of the research conducted for this thesis, in this publication we present results from experiments with an initial version of our **hybrid system combination** framework.

2.5 Quality Evaluation

[**Papineni et al., 2002**] **BLEU** is the de-facto standard for all machine translation evaluation campaigns. It is based on *n*-gram overlap statistics with one or several references.

[**Doddington, 2002**] This paper defines the so-called **NIST** score for automatic evaluation of machine translation quality. It is based on co-occurrence statistics on the *n*-gram level and derived from BLEU.

[**Denkowski and Lavie, 2011**] The authors describe **Meteor**, an automatic metric for reliable optimisation and evaluation of machine translation output. Meteor scores candidate translations by aligning them to one or several references. Using these alignment links, the metrics computes overlap with the reference data. Meteor has proven to be an evaluation metric which produces reliable scores on the segment level.

[**Aziz et al., 2012**] This paper describes the design and implementation of **PET**, a tool for post-editing and the qualitative assessment of machine translation output. The authors' main focus lies in post-editing of MT output. As with many recent tools, the software and its source code are freely available.

[**Dreyer and Marcu, 2012**] The authors describe **HyTER**, a method which performs machine translation evaluation based on meaning-equivalent semantics. HyTER is capable of compactly encoding an exponential number of correct translations. Potentially, this allows to better cope with ambiguities in the translation process.

[**Federmann, 2012b**] This paper presents **Appraise**, an open-source toolkit for manual evaluation of machine translation quality. Appraise has been developed as part of the research conducted for this thesis and has become the official evaluation system in WMT 2013.

2.6 Machine Learning

[Rosenblatt, 1958] A fundamental publication introducing the **perceptron** algorithm which can be used for incrementally learning a model which is able to perform linear classification.

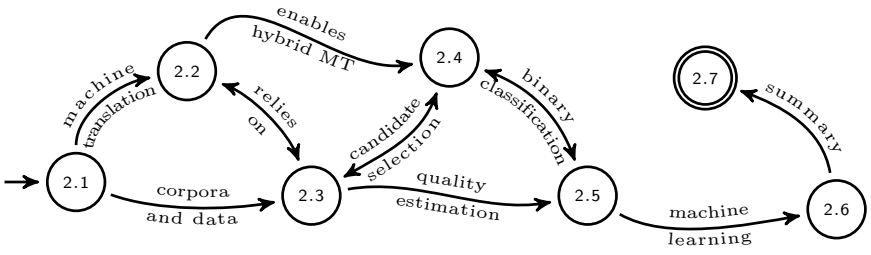
[Vapnik, 1995] This seminal work describes **support vector machines** which can be used to train binary classification models, non-linearly mapping from input vectors to a very high-dimension feature space in which a linear decision surface can be constructed.

[Tipping, 2001] An alternative to the aforementioned SVMs, the **relevance vector machine** is of comparable classification quality. In theory, it is able to achieve this with fewer support vectors (or relevance vectors, as they are called in this approach). RVMs represent a theoretical alternative for support vector machines.

[Fan et al., 2008] The authors discuss **liblinear** which is a library for large linear classification. For linear machine learning problems which can be tackled without using kernel functions and the infamous kernel trick, liblinear provides an extremely efficient solution, typically outperforming libSVM.

[Chang and Lin, 2011] This paper presents **libSVM** which is a popular toolkit implementing support vector machines as introduced in [Vapnik, 1995].

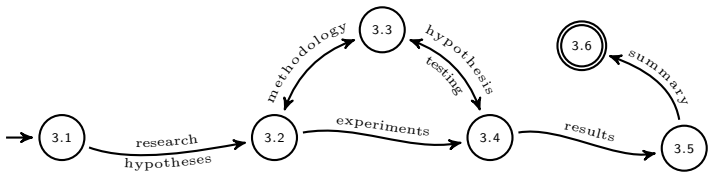
[Pedregosa et al., 2011] The authors describe **scikit-learn**, a Python-based framework for machine learning. It provides wrappers to, e.g., libSVM and liblinear and also takes care of, e.g., data normalisation and cross validation. We use scikit-learn in our experiments for the verification of our research hypotheses.



2.7 Chapter Summary

In this chapter we have briefly summarised relevant background literature which has inspired and influenced the research work we have conducted for this thesis. Additional review of the state of the art has already been given throughout introductory Chapter 1. We have provided a chronological overview on selected relevant publications from the fields of 1) machine translation, 2) corpora and data sets, 3) hybrid combination systems, 4) quality evaluation, and 5) machine learning.

In the next chapters we describe how these ingredients can be combined to yield an effective framework for system combination based what we introduce as joint, comparison-based binarisation of feature vectors. These can be used to estimate binary classification models for sentence selection.



3

System Combination using Optimal Sentence Selection

"A University should be a place of light, of liberty, and of learning."
– **Benjamin Disraeli**: speech before the House of Commons, March 11, 1873.

"Nothing great was ever achieved without enthusiasm."
– **Ralph Waldo Emerson**: *Essays*, 1841. 'Circles'.

3.1 Introduction

As we have seen, system combination of machine translation output can be addressed in multiple ways. We have previously focused on two different combination techniques: 1) *confusion networks* and 2) *sentence selection*. The first has seen extensive research work in recent years, most notably during 2009–2011 when the yearly Workshop on Statistical Machine Translation (WMT) offered a dedicated system combination task. It is clear that confusion network decoding achieves good performance w.r.t. output translation quality. Interestingly, the second approach has remained largely neglected in machine translation research. We investigate the potential performance gain attained by selection among a set of different candidate translations in the remainder of this chapter.

Problem Statement

Let us first define the specific problem we are investigating:

> *There seems to be a general preference towards using confusion network decoding for system combination. Is this forever carved in stone? Can we actually use sentence selection approaches for system combination?*

While there has been some limited research on the application of sentence selection approaches for system combination [Hildebrand and Vogel, 2008, Hildebrand and Vogel, 2009, Hildebrand and Vogel, 2010], we wanted to find empirical proof that such methods could be able to outperform translation quality of the respective *source systems* on real data. It is clear that any method for system combination which only integrates *better* candidates on the sentence level, in theory, monotonously improves overall translation quality. The interesting question is whether data collected in real application scenarios such as, e.g., the yearly WMT shared tasks supports our initial assumption that there are good candidate translations contained in globally bad systems. For this, we aim to measure how much of an improvement in terms of translation quality can be observed if one was able to perform optimal classification of candidate translations. To the best of our knowledge, there exists no such study in the literature.

This chapter is structured as follows: in Section 3.2 we state our research hypotheses. Second, we describe the methodology (Section 3.3) we have used in our experiments which are discussed afterwards (Section 3.4). Finally, we present results (Section 3.5) and some observations before ending with a summary of our findings and a conclusion in Section 3.6.

3.2 Research Hypotheses

In order to make sense as a methodological paradigm for addressing system combination problems, we have to check whether sentence selection methods can theoretically result in an improved overall translation quality, compared to the quality of the individual source systems from which we synthesise the combined translation. Hence, our first working hypothesis is:

Hypothesis 1. *Sentence selection methods can outperform their source systems on real data.*

While this is trivial from a theoretical point of view (especially when considering that our perfect oracle-based section can only improve over the single-best system as it will only choose other candidate translations if they are locally optimal, i.e., better than the single-best translation) the interesting question is whether independent data collected over time shows that (empirically) the sentence-based combination of several candidate translations outperforms the single-best system. This basically answers the question whether globally bad systems can contribute to an overall improvement in terms of translation quality.

Second, we intend to investigate whether sentence selection depends on external factors such as the underlying technological paradigm of the individual candidate systems or the language pair under investigation. It seems obvious that this should not be the case, however we have yet to see if we can find empirical proof for such claim.

Hypothesis 2. *Sentence selection approaches show improvement potential across language pairs or underlying technological paradigms of the source systems.*

Finally, we want to find out whether translation systems which perform bad on the overall system level can be beneficial in the context of sentence selection. Again, the initial assumption is that they can likely contribute something to our final combination results. If and how much remains to be seen in our experiments. Our final working hypothesis is:

Hypothesis 3. *Even systems which perform bad on the global system level can contribute helpful segment translations.*

3.3 Methodology

Definitions

Averaged Meteor

We have already indicated our preference for the Meteor evaluation metric[1] as it has shown a better correlation with human judgment in our experiments over the course of several years (see, e.g., [Federmann, 2011]). The computation of Meteor scores on the system level is effectively based on the performance on both document and segment level.

Notation 1. *We denote the Meteor score on the segment level as* $\boldsymbol{Meteor}_{segment}$.

In order to speed up computation in our selection experiments, we choose to define a new variant of Meteor which we call "averaged Meteor". As the name indicates, an averaged Meteor score for the system level is computed by averaging the individual segment scores for the test set.

Formally, we define this as follows:

[1] We used Meteor v1.4 which is available from `http://www.cs.cmu.edu/~alavie/METEOR/`.

Definition 2 (Averaged Meteor)**.** *We compute an* ***averaged Meteor*** *score by averaging the scores of all N segments contained in the current test set:*

$$Meteor_{\text{AVG}}(\{segment_1, \ldots, segment_N\}) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{N} Meteor(segment_i) \qquad (3.1)$$

Notation 2 (Meteor$_{\text{AVG}}$)**.** *We denote the averaged Meteor score as* ***Meteor***$_{\text{AVG}}$*.*

In practice, the use of an averaged Meteor score can result in a slight overestimation of the real Meteor score as it would have been computed by the Meteor tool on the same data. This is caused by the fact that we ignore factors such as sentence length and the fragmentation penalty. It does, however, not have too big an influence on the final scores (*and it speeds up computation*) and is hence neglected in our research work. Michael Denkowski, one of the authors of the Meteor score, stated in an email:

> "Meteor computes a corpus-level score using the same formula for sentence-level scoring. Subject to a slight approximation, which has to do with one special case in the fragmentation penalty, this is equivalent to the average of sentence-level scores."
>
> – **Michael Denkowski**: *Email*, August 21, 2013.

Given a large corpus of source text and the corresponding translations produced by multiple machine translation systems, we want to estimate how much of an improvement in terms of translation quality (as measured by our freshly defined Meteor$_{\text{AVG}}$ score) we can obtain by performing *optimal sentence selection*. This is, of course, a theoretical *gedankenexperiment* as we are only focused on the method's performance without implementing the process that would compute Meteor scores (or rather *estimate* them given that we would not have reference text in application scenarios) on unseen data. We assume that we can compute Meteor$_{segment}$ scores and perform our

experiments on data sets for which we have access to the corresponding reference texts.

Definition 3 (Optimal Sentence Selection)**.** *Given a set of multiple candidate translations* $C = \{candidate_1, \ldots, candidate_K\}$*, with cardinality* $K \geq 2$*, for the same source segment* S*, we perform* ***optimal sentence selection*** *by selecting the candidate* c_i *which maximises the segment-level Meteor score as computed by* $Meteor_{segment}$*. Formally:*

$$Select(C) \stackrel{\text{def}}{=} \underset{c \in C}{\operatorname{argmax}} Meteor_{segment}(c) \tag{3.2}$$

Combination with Optimal Classification

The basic idea of our experiments in this chapter can be summarised in the following way:

1. Compute standard Meteor scores for all candidate translations. This will result in scores on 1) system, 2) document, and 3) segment level;
2. Sort candidate systems according to *some order*;
3. Given the ordered set of N candidate systems, compute all $N-1$ *"k-best"* combinations and measure their respective $\text{Meteor}_{\text{AVG}}$ scores;
4. Compute potential performance gain by comparing resulting scores to the single-best candidate system score.

Note how we use the admittedly fuzzy term *"some order"*: essentially, our combination method works on an ordered set of N candidate systems. For these, we compute the $N-1$ *"k-best"* combinations where "better" refers to the order of the set of candidate systems, e.g., candidate_1 is assumed to be "better" than candidate_2 and so on. We apply three different combination strategies which are described next.

Top-k Combination

As the name implies, the **top-k combination** strategy aims at combining translation systems ordered "best to worst" by their respective system-level Meteor translation quality. As our optimal selection can only ever change a segment translation if some other system has a better local segment score, this combination approach is guaranteed to always outperform the single-best translation system (which is Top-k_1). Remember that we are assuming *optimal classification* to be available in this chapter. In real experiments without access to reference texts, segment-level Meteor scores have to be estimated in some way. We come back to that later in this thesis.

Given a set of candidate systems $x_i \in Systems$, we define the (ordered) set of top-k systems (for $2 \leq k \leq N$) in the following way:

$$\text{Top-}k \stackrel{\text{def}}{=} \{x_1, \ldots, x_k \,|\, Meteor_{\text{AVG}}(x_i) \geq Meteor_{\text{AVG}}(x_j) \,\forall 1 \leq i < j \leq k\} \tag{3.3}$$

This combination method should enable us to verify Hypothesis 1.

Worst-k Combination

Conversely, we define the **worst-k combination** strategy for systems $x_i \in Systems$ which does allow to investigate how many good segments can be found within the subset of translation systems which perform bad on the system level, effectively seeking empirical proof for Hypothesis 3.

$$\text{Worst-}k \stackrel{\text{def}}{=} \{x_1, \ldots, x_k \,|\, Meteor_{\text{AVG}}(x_i) \leq Meteor_{\text{AVG}}(x_j) \,\forall 1 \leq i < j \leq k\} \tag{3.4}$$

Note that due to the fact that we start with the two worst candidate systems according to the system-level Meteor score, this combination approach need not outperform the single-best translation system for small values of k. The addition of the final system (Worst-k_N which is actually Top-k_1) should at least result in this system's level of performance.

Algorithm 2 Optimal sentence selection for N candidate systems

Require: set of translations from N candidate systems $S = \{S_1, S_2, \ldots, S_N\}$.
Ensure: $|S_1| = |S_2| = \ldots = |S_N|$

1: $T' \leftarrow \emptyset$

2: **for each** segment id $i, 1 \leq i \leq$ # of segments **do**
3: segment-translations $\leftarrow \{S_{1,i}, \ldots, S_{N,i}\}$ ▷ Extract translations for current segment id
4: $T' = T' \cup Select$(segment-translations) ▷ Select by maximising Meteor$_{segment}$ score
5: **end for**

6: **return** T' ▷ Return combined translation output

Alternate-k Combination

Third, we define a so-called **alternate-k combination** approach. For this, we alternate between good and bad translation systems to see how such a collection of systems can perform. As is the case for the Top-k strategy, this method has a worst-case lower-bound of Meteor$_{\text{AVG}}$(Top-k_1).

$$Alternate\text{-}k \stackrel{\text{def}}{=} \begin{cases} \{Top\text{-}k_1, Worst\text{-}k_1, \ldots, Worst\text{-}k_{\lfloor N/2 \rfloor}\} & \text{if N even} \\ \{Top\text{-}k_1, Worst\text{-}k_1, \ldots, Top\text{-}k_{\lceil N/2 \rceil}\} & \text{else} \end{cases} \tag{3.5}$$

Algorithm 2 illustrates how we can compute a combined translation from a set of N candidate translations using optimal sentence selection. We iterate over the set of segments and select the candidate translation maximising the local Meteor$_{segment}$ score for the current segment under investigation. We show an implementation of this algorithm in form of Python code in listing 3.1. It requires a set of two or more translation systems as `system_ids` in *some* order as input. We assume a global dictionary `SYSTEM_SCORES` which, for each system (as identified by the respective `system_id`) maps from all individual segment identifiers to the corresponding segment scores.

Listing 3.1: Python implementation of sentence selection

```
def combine_systems(system_ids):
    """
    Combines systems by maximising segment-level Meteor score.

    Global dictionary SYSTEM_SCORES maps from segment id to score.
    """
    if len(system_ids) < 2:
        return compute_meteor_avg(system_ids[0])

    # Create mutable copy of the first system's segment scores.
    data = SYSTEM_SCORES[system_ids[0]].copy()
    for key, value in data.items():
        best_value = value
        for other_system in system_ids[1:]:
            try:
                other_value = SYSTEM_SCORES[other_system][key]
                if other_value > best_value:
                    best_value = other_value

            except KeyError:
                continue

        if best_value > value:
            data[key] = best_value

    segment_scores = data.values()
    del data
    return sum(segment_scores) / len(segment_scores)
```

3.4 Experiments

In order to investigate the potential performance gains attained by optimal sentence selection, we need a large corpus of translations including the corresponding reference text. This corpus needs to contain translation output generated by a plethora of individual MT systems, implementing various technological paradigms. Furthermore, we require that this corpus and its individual translations have been collected over the course of several years by an independent party to avoid bias.

Data

Luckily, such data exists as it is released, year after year, as part of the Workshop on Statistical Machine Translation (WMT). We take the data packages from 2007 until 2013 and build a joint research corpus from these.[2] Additionally, we make use of the data we have collected during the ML4HMT Workshops in 2011 and 2012 [Federmann, 2011, van Genabith et al., 2012]. We will provide a brief overview on the individual data sets below.

WMT 2007

Data for WMT 2007 has been produced from March 23 until April 6, 2007. The workshop featured a shared translation task with two different test sets: *test2007* which contains 2,000 sentences taken from EuroParl [Koehn, 2005], and *nc-test2007* which features a total of 2,007 sentences collected from the news commentary domain. The shared task covered the following languages: Czech, German, Spanish, French, and English. For more information refer to official results summary paper [Callison-Burch et al., 2007].

[2] We intend to make our joint corpus publicly available at LREC 2014.

WMT 2008

Data for WMT 2008 has been produced from March 14 until March 21, 2008. The workshop again featured a shared translation task, this time with three different test sets: *test2008* which contains 2,000 sentences from the EuroParl corpus, *nc-test2008* featuring 2,028 sentences taken from the news commentary domain, and finally *newstest2008* with a size of 2,051 sentences extracted from major news outlets such as the *BBC*, *Der Spiegel*, or *Le Monde*. The shared task added Hungarian and a non-English language pair, namely Spanish↔German in both translation directions. Official workshop results are available from [Callison-Burch et al., 2008].

WMT 2009

Data for WMT 2009 has been produced from December 8 until December 12, 2008. The workshop added a dedicated system combination task which took place from December 22, 2008 until January 5, 2009. Both tasks focused on news translation as the only test set: *newstest2009* contains 2,525 sentences from the news domain. The shared task covered the following languages: Czech, German, Spanish, French, Hungarian, and English. The combination task investigated performance of combinations of the translations which had been generated in the translation task. The official results summary paper is available from [Callison-Burch et al., 2009].

WMT 2010

Data for WMT 2010 has been produced from March 1 until March 5, 2010. The workshop once again featured a system combination task and focused on news translations as the single test set: *newssyscombtest2010* which contains exactly 2,034 sentences from the news domain. Both shared tasks covered the

following languages: Czech, German, Spanish, French, and English, however dropping Hungarian which was part of previous workshops. By contrast to previous editions of the workshop series, WMT 2010 substantially increased the amount of available training materials. For more information refer to the official overview paper which is available from [Callison-Burch et al., 2010].

WMT 2011

Data for WMT 2011 has been produced from March 14 until March 20, 2011. The workshop for the third time featured a system combination task. Also, there was a so-called *featured translation task*, in which participants worked on translating Haitian Creole SMS messages (collected in the aftermath of the 2010 Haitian earthquake) into English. As was the case for previous editions, it focused on translation of news stories. The single test set, *newstest2011*, contains 3,003 sentences from this domain. The shared task covered these languages: Czech, German, Spanish, French, and English. The official results paper is available from [Callison-Burch et al., 2011].

WMT 2012

Data for WMT 2012 has been produced from February 27 until March 2, 2012. Next to the usual translation task, the workshop feature a dedicated metrics task as well as a shared task on quality estimation without reference text. The translation task focused on news translations as the single test set: *newstest2012* contains 3,003 sentences. It covered the following languages: Czech, German, Spanish, French, and English. For more details refer to the workshop summary paper which is available as [Callison-Burch et al., 2012].

WMT 2013

Data for WMT 2013 has been produced from April 29 until May 3, 2013. The workshop continued all three shared tasks from the previous year, namely translation, metrics, and quality estimation task. The translation task focused a single test set: *newstest2013* consists of 3,000 sentences take from the news domain. Next to Czech, German, Spanish, French, and English, the shared task added Russian as an additional language. Manual evaluation has been implemented using *Appraise* which is a contribution of this thesis. Official results are available in [Bojar et al., 2013].

ML4HMT

Data for the ML4HMT workshops in 2011 and 2012 has been produced from May 20 until October 10, 2011 and from August 23 until October 28, 2012, respectively. Test sets were adapted from previously released WMT data sets, namely *newstest2008*, from which the final 1,026 sentences are used, for ML4HMT 2011 and *newstest2011* containing 3,003 sentences for ML4HMT 2012. The shared tasks covered the Spanish→English language pair. For more information see [Federmann, 2011, van Genabith et al., 2012].

Selection Experiments				
Language pair	Systems	Combinations	Sentences	Words
Czech→English	56	144	144,357	3,063,615
German→English	149	423	372,753	8,273,536
Spanish→English	132	363	327,217	7,435,008
French→English	148	417	363,775	8,285,114
Hungarian→English	3	6	7,575	169,034
Any→English	3	6	6,102	135,559
English→Czech	81	213	205,357	3,610,335
English→German	140	396	358,276	7,374,138
English→Spanish	120	339	298,168	6,964,052
English→French	132	378	329,878	7,755,776
Total	964	2,685	2,413,458	53,066,167

Table 3.1: Statistics for optimal sentence selection experiments

3.5 Results

General Overview

Table 3.1 summarises the data used in our experiments with optimal sentence selection. It details the number of 1) candidate systems, 2) combinations tested, 3) total sentences, and 4) total words per language pair. Except for language pairs Hungarian→English and Any→English, we have tested a large amount of systems and corresponding combinations. In total, we have experimented with 964 candidate systems and 2,685 combinations. A detailed inspection of our observations can be found on the following pages.

Our main interest has been the verification of our research hypotheses as defined earlier in this chapter (Section 3.2). Based on our experiments, we can confirm that Hypothesis 1, *Sentence selection methods can outperform their source systems on real data.*, holds. In fact, we are able to report improvements for all language pairs, test set, and data set combinations. While this has been expected, we now have found empirical evidence for this statement.

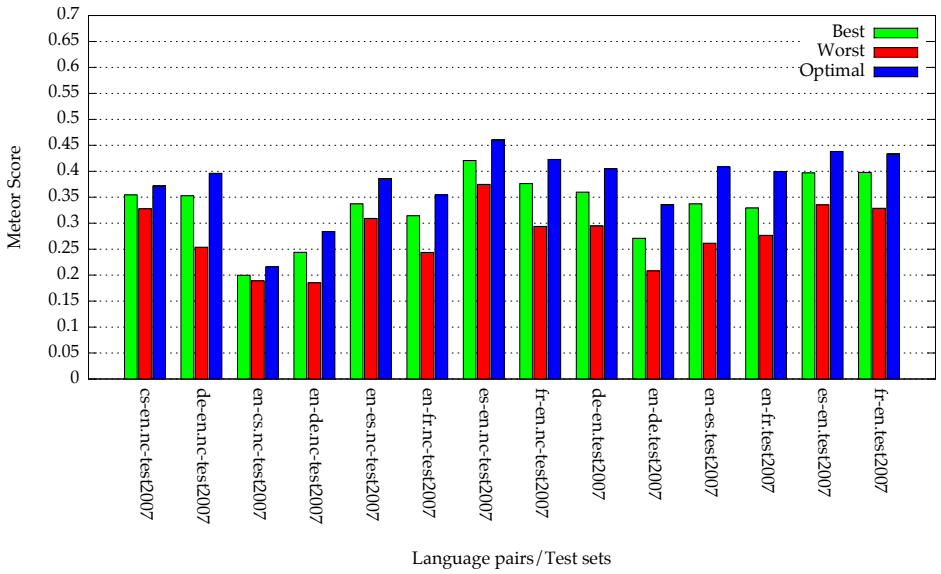


Figure 3.1: Sentence selection performance for WMT 2007 data.

WMT 2007

Figure 3.1 shows the results for our sentence selection experiments with data from WMT 2007. Optimal selection does indeed outperform the single-best translation system, irrespective of test set or language pair. Test set *test2007* sees an average potential relative gain of +18.9% compared to the "Best" system while second test set *nc-test2007* achieves +11.3%. Note how English→Czech performs much worse (with around 0.2 $\text{Meteor}_{\text{AVG}}$ score) than the inverse direction (scoring around 0.35). Translation into German seems to be difficult and results in only moderate scores in the 0.25 range. For both test sets, translation quality of language pair English→German could be drastically improved by performing an optimal sentence selection approach, to 0.33 for test set *test2007*, a +23.9% increase, and to 0.28 for test set *nc-test2007*, a +16.3% jump in terms of translation quality. We observe the overall biggest performance gain for test set *nc-test2007* for language pair Spanish→English where we measure an improvement of +22.3%.

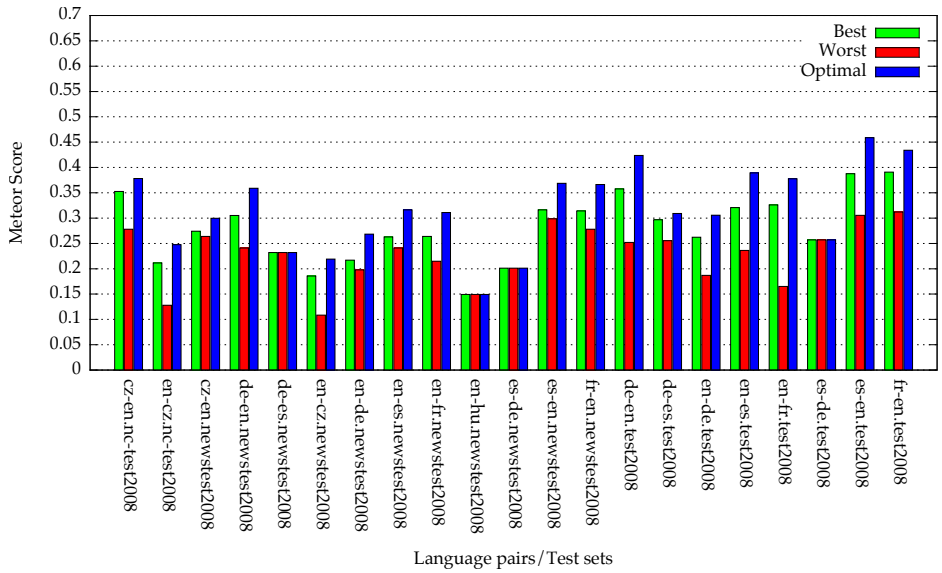


Figure 3.2: Sentence selection performance for WMT 2008 data.

WMT 2008

Figure 3.2 gives the results for our experiments with data from WMT 2008. Again, we are able to observe an improvement for all language pairs and test sets. Test set *test2008* achieves an average gain of +17.0%, test set *nc-test2008* sees an improvement of +12.2%, while test set *newstest2008* reports an average of +17.5%. The fact that English→Hungarian does not show any positive change is related to the fact that only a single candidate system generated translation output for this translation direction. Translation quality of this language pair is around 0.15 $\text{Meteor}_{\text{AVG}}$ score; we are confident that our combination approach would outperform this baseline provided an additional translation was available. Language pair Spanish→German fails to report any improvement for the same reasons. Similar to our findings from the previous workshop, language pairs English→Czech as well as English→German achieve low scores when compared to the other language pairs. The biggest jump in performance gain can be observed for language pair English→German and test set *newstest2008*, a +21.3% increase.

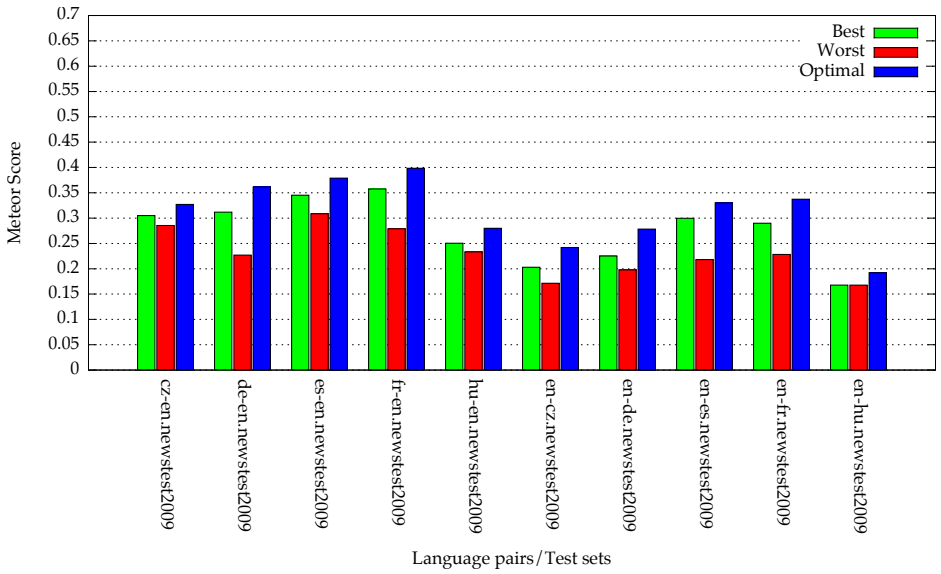


Figure 3.3: Sentence selection performance for WMT 2009 data.

WMT 2009

Figure 3.3 shows the results for our sentence selection experiments with data taken from the WMT 2009 workshop. This workshop was the first to feature only one test set, namely *newstest2009* for which we achieve an average improvement of +14.0%. Translation from English into complex languages such as Czech and Hungarian remain difficult as can be seen from the comparably low $\text{Meteor}_{\text{AVG}}$ scores, around 0.2 for English→Czech and around 0.17 for English→Hungarian. By contrast to WMT 2008, this time we are able to report a performance gain for the latter language pair also. Interestingly, both individual source systems perform roughly on the same level of translation quality (see how "Best" and "Worst" bars are identical). However, their combination outperforms this non-ambiguous baseline. The overall biggest improvement in terms of $\text{Meteor}_{\text{AVG}}$ score can be observed, once again, for language pair English→German. In 2009, a giant increase of +23.9% can be measured which is among the biggest gains observed in our experiments.

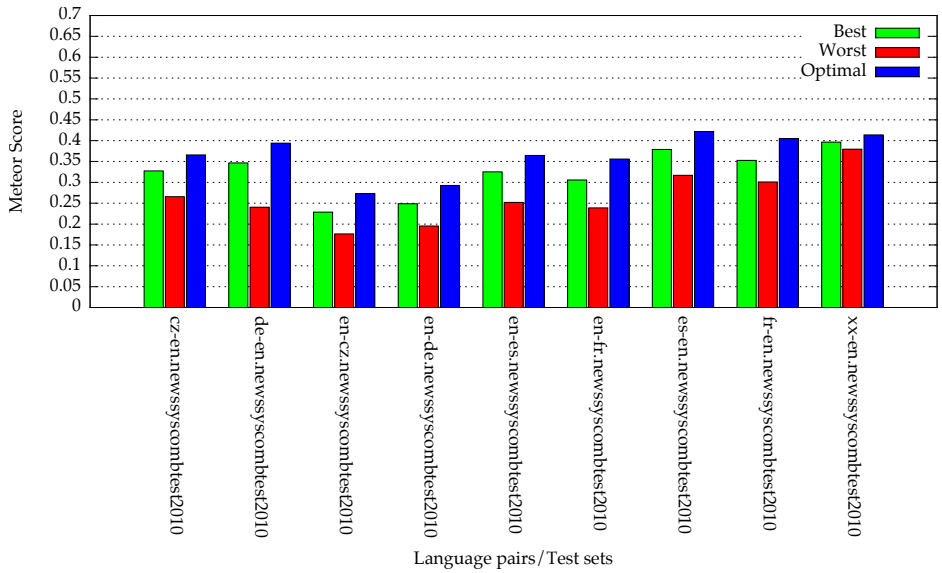


Figure 3.4: Sentence selection performance for WMT 2010 data.

WMT 2010

Figure 3.4 depicts the results from our experiments with optimal sentence selection on data derived from WMT 2010. Again, there exists only a single test set which is called *newssyscombtest2010*. As the name implies, it is used for both the shared translation task and the system combination task as well. The average performance gain by optimal selection is measured as +13.5%. Across languages pairs, we observe improved scores in the 0.3 $\text{Meteor}_{\text{AVG}}$ range, except for notoriously difficult language pairs English→Czech (around 0.22) and English→German (around 0.25). Judged by these numbers, it seems that machine translation quality has seen a boost in WMT 2010. The biggest jump in performance gain due to optimal sentence selection can be measured for language pair English→Czech, an improvement of +19.5%. Language pair Any→English (labeled as xx-en in the figure) represents a meta-combination using results from the system combination task as candidate systems. The resulting improvement is modest at around +4.29% which might be caused by the small number of candidate systems for this task.

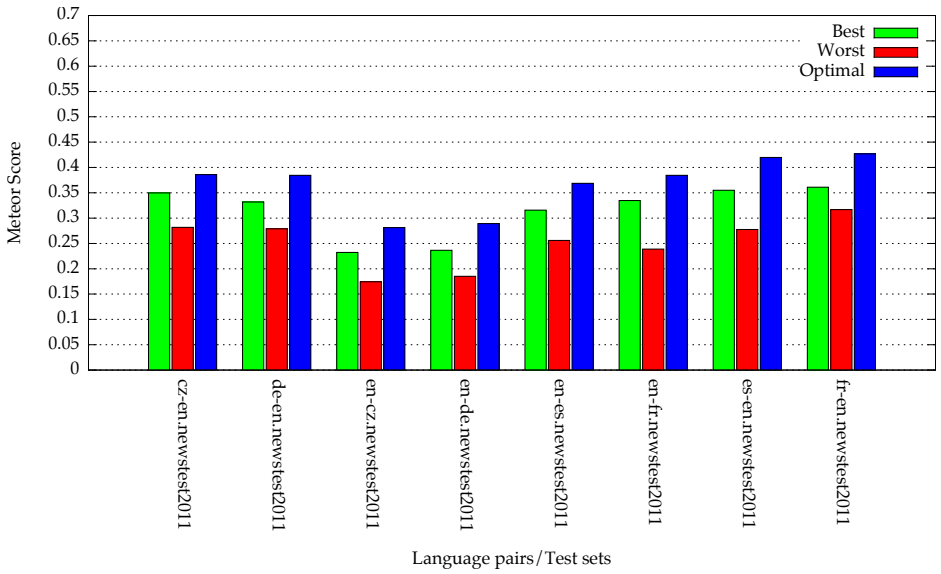


Figure 3.5: Sentence selection performance for WMT 2011 data.

WMT 2011

Figure 3.5 shows results from our sentence selection experiments on data from WMT 2011. The single test set is called *newstest2011*. The average performance gain which is attained using optimal selection is measured as +17.2%, an increase of +27.4% in comparison to 2010 with an average gain of +13.5%. Improved scores reach the 0.37–0.43 $\text{Meteor}_{\text{AVG}}$ range for most language pairs, except English→Czech (around 0.28) and English→German (around 28.9). As was the case in the previous year, we are able to observe a jump in translation quality of the given candidate systems and, thus, an increase in potential gain by optimal sentence selection. The largest gain can be observed for language pair English→German, seeing an increase of 22.3%. English→Czech is very close with an improvement of +21.1%. It seems that both language pairs are still more difficult than other language pairs which correlates to their huge potential gains. It is also noteworthy that the qualitative spectrum of participating translation systems has widened; the average difference between the best and the worst systems has grown.

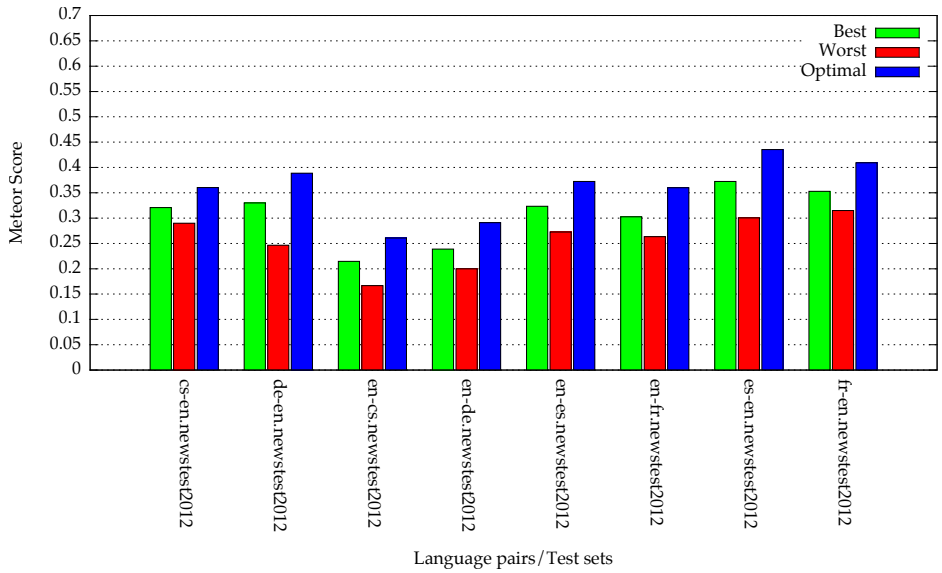


Figure 3.6: Sentence selection performance for WMT 2012 data.

WMT 2012

Figure 3.6 shows the results for our sentence selection experiments with data taken from the WMT 2012 workshop. We see that (similar to previous years) language pairs English→Czech and English→German remain the hardest language pairs with lowest corresponding Meteor scores. This workshop again features only one test set, namely *newstest2012* for which we achieve an average improvement of +17.6%. The lowest performance gain is observed for language pair Czech→English at 0.36 (a relative improvement of +12.3%). The overall biggest improvement in terms of $\text{Meteor}_{\text{AVG}}$ score can be observed, once again, for language pair English→German. In 2012, we measure an increase of +21.8%, resulting a Meteor score around 0.29.

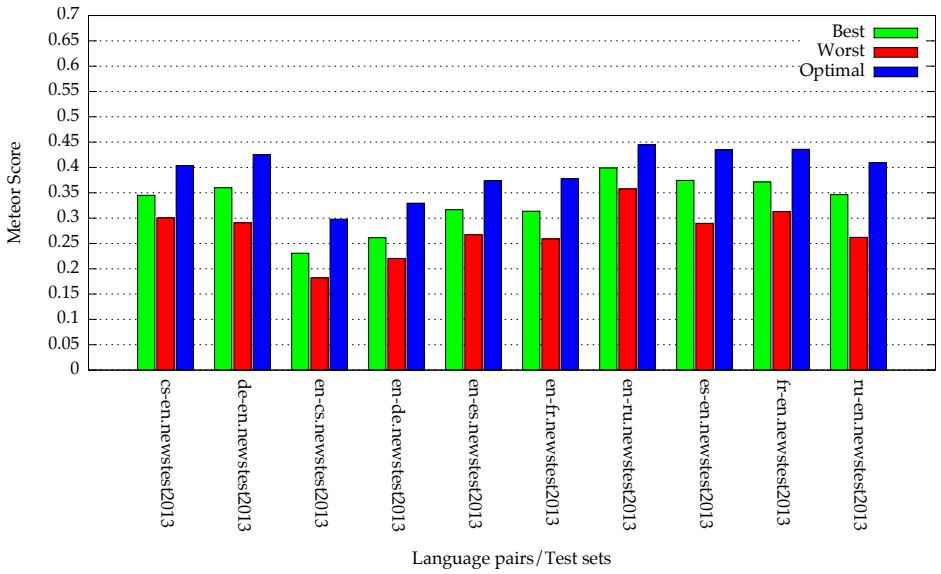


Figure 3.7: Sentence selection performance for WMT 2013 data.

WMT 2013

Finally, Figure 3.7 lists the results for our sentence selection experiments with data taken from the WMT 2013 workshop. Once again, language pairs English→Czech and English→German remain the hardest language pairs with lowest corresponding Meteor scores, Russian is added to the shared translation task as a new language. This workshop features a single test set, namely *newstest2013* for which we achieve an average improvement of +19.2%. The lowest performance gain is observed for English→Russian at 0.45 (a relative improvement of +11.5%). The corresponding single-best score at around 0.40 is already pretty strong, which explains that no bigger improvement can be achieved. The overall biggest improvement in terms of $\text{Meteor}_{\text{AVG}}$ score can be observed for language pair English→Czech. In 2013, we measure an increase of +29.2%, resulting a Meteor score around 0.30.

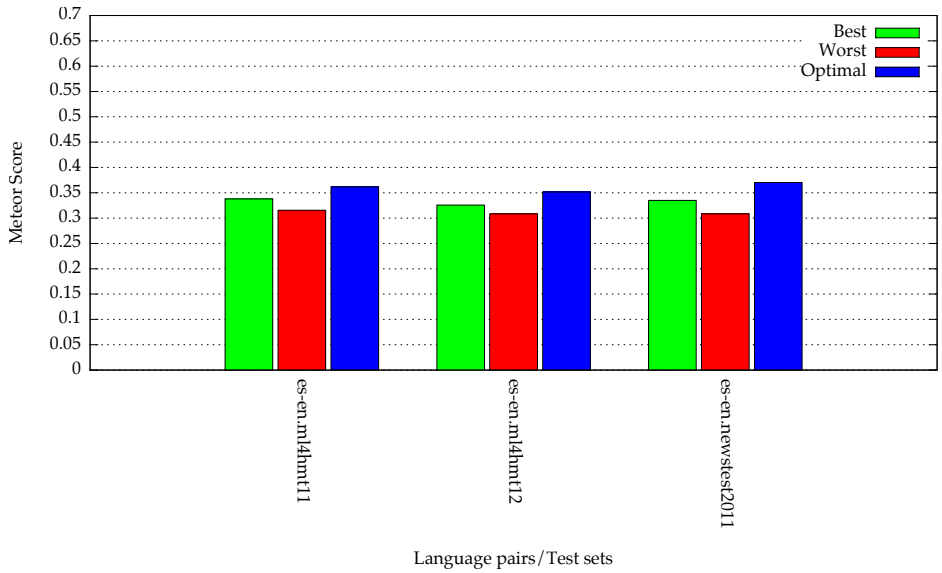


Figure 3.8: Sentence selection performance for ML4HMT data.

ML4HMT

Figure 3.8 shows the results we have obtained with data taken from the ML4MT shared tasks. Test set *ml4hmt11* achieves an average improvement of +7.07% and test set *ml4hmt12* performs slightly better with a potential gain of +7.89%. The third test set, namely *newstest2011*, does include the original translations which have been used by shared task participants to build their systems. It achieves an average performance boost of +10.5%. This allows to draw two conclusions. First, it becomes clear that combined translations from ML4HMT 2012 did not optimally integrate information made available by the supplied training materials. Second, it becomes apparent that the best combination system did not outperform the single-best source system. It seems that pre-dominant use of confusion networks has resulted in a loss of information. Interestingly, the only system implementing sentence selection achieves the best Meteor_{system} score. This supports our initial assumption that sentence selection can outperform confusion networks.

Dependency on Languages

So far, we have already seen that sentence selection approaches can outperform the single-best candidate system. We now want to analyse our results with a special focus on the language pair under investigation.

Czech→English

Table 3.2 provides details of our results for language pair Czech→English. We can see that optimal sentence selection results in stable improvements for all individual test sets. The biggest performance gain has been measured for test set *newssyscombtest2010*, an increase of +11.7%. Note that this is also the test set with most candidate systems which seems to beneficial to our method due to the increased inventory of candidate translations to choose from. The average improvement is +8.05%.

German→English

Table 3.3 shows the results for language pair German→English. Again, we are able to observe improvements across all test sets. By contrast to previous language pair Czech→English, the performance gains are larger, measuring from +12.2% for test set *nc-test2007* up to +18.5% for test set *test2008*. The average performance gain is +15.1%.

Spanish→English

Table 3.4 depicts results for language pair Spanish→English. As this is the only language pair which is featured in the ML4HMT shared tasks, we have the largest overall number of test sets across language pairs. Performance gains range from +7.07% for test set *ml4hmt11* up to +22.3% for test set *test2007*. The average improvement is +12.6%.

French→English

Table 3.5 gives detailed results for experiments with data from language pair French→English. Consistent improvements can be observed, ranging from +11.1% for test set *test2008* to +16.7% for test set *newstest2008*. The overall average performance gain is +13.1%.

Hungarian→English

Table 3.6 provides an overview on language pair Hungarian→English. Only one test set exists for this language pair, namely *newstest2009*, for which we observe an improvement of +11.7%.

Russian→English

Table 3.13 gives results for language pair Russian→English. As this language pair has only been used as part of WMT 2013, there exists only one test set (*newstest2013* with an overall performance gain of +11.5%.

Any→English

Table 3.7 shows results from the meta-combination task for the Any→English "language pair". There is only a single test set available, *newssyscombtest2010*, which achieves a performance gain of +4.29%.

English→Czech

Table 3.8 reports optimal sentence selection performance for language pair English→Czech. We see improvements across all test sets, ranging from +7.95% for test set *nc-test2007* up to +19.5% for test set *newssyscombtest2010*. The overall average gain is +16.4%.

English→German

Table 3.9 depicts results for language pair English→German. Improvements are massive, ranging from +16.3% for test set *nc-test2007* to +23.9% for test set *test2007*. This is the best performing language pair in our experiments, reaching an average performance gain of +20.3%.

English→Spanish

Table 3.10 gives an overview on results for language pair English→Spanish. Performance gains are consistent across test sets, measuring from +10.1% for test set *newssyscombtest2010* up to +21.3% for test set *test2008*. The average improvement is +16.6%.

English→French

Table 3.11 provides a detailed look on our experiments for language pair English→French. Improvements range from +12.7% for test set *nc-test2007* to +21.3% for test set *test2007*. The average performance gain is +16.7%.

English→Russian

Table 3.12 shows results for language pair English→Russian which has been added as part of WMT 2013. The average improvement is +18.2%

Czech→English				
Test set	Systems	Best	Worst	Optimal
nc-test2007	3	0.354	0.328 (-7.48%)	0.371 (+4.82%)
nc-test2008	4	0.353	0.278 (-21.1%)	0.378 (+7.23%)
newstest2008	3	0.274	0.264 (-3.80%)	0.300 (+9.27%)
newstest2009	3	0.305	0.286 (-6.34%)	0.327 (+7.24%)
newssyscombtest2010	13	0.328	0.266 (-18.9%)	0.366 (+11.7%)
newstest2011	12	0.350	0.282 (-19.4%)	0.386 (+10.3%)
newstest2012	6	0.321	0.290 (-9.65%)	0.360 (+12.3%)
newstest2013	12	0.345	0.300 (-13.0%)	0.404 (+17.1%)
Average	5	0.323	0.284 (-11.5%)	0.348 (+8.05%)

Table 3.2: Test set performance for language pair Czech→English

German→English				
Test set	Systems	Best	Worst	Optimal
nc-test2007	5	0.353	0.254 (-28.1%)	0.396 (+12.2%)
test2007	7	0.360	0.295 (-18.2%)	0.405 (+12.4%)
newstest2008	13	0.305	0.241 (-21.0%)	0.359 (+17.7%)
test2008	14	0.358	0.252 (-29.5%)	0.424 (+18.5%)
newstest2009	15	0.312	0.227 (-27.2%)	0.361 (+15.9%)
newssyscombtest2010	31	0.347	0.241 (-30.6%)	0.394 (+13.7%)
newstest2011	26	0.332	0.279 (-15.9%)	0.384 (+15.9%)
newstest2012	16	0.330	0.247 (-25.3%)	0.389 (+17.8%)
newstest2013	23	0.360	0.291 (-19.3%)	0.425 (+18.0%)
Average	16	0.340	0.258 (-23.9%)	0.393 (+15.8%)

Table 3.3: Test set performance for language pair German→English

Spanish→English				
Test set	Systems	Best	Worst	Optimal
ml4hmt11	4	0.338	0.315 (-6.61%)	0.362 (+7.07%)
ml4hmt12	6	0.326	0.308 (-5.37%)	0.352 (+7.89%)
newstest2011	10	0.335	0.308 (-7.94%)	0.370 (+10.5%)
nc-test2007	7	0.421	0.375 (-10.9%)	0.460 (+9.39%)
test2007	8	0.399	0.335 (-15.9%)	0.488 (+22.3%)
newstest2008	13	0.316	0.299 (-5.54%)	0.369 (+16.6%)
test2008	15	0.388	0.305 (-21.2%)	0.459 (+18.4%)
newstest2009	9	0.345	0.309 (-10.6%)	0.379 (+9.72%)
newssyscombtest2010	16	0.379	0.317 (-16.4%)	0.422 (+11.3%)
newstest2011	16	0.355	0.278 (-21.8%)	0.420 (+18.3%)
newstest2012	12	0.372	0.301 (-19.3%)	0.435 (+16.9%)
newstest2013	17	0.375	0.290 (-22.7%)	0.434 (+16.0%)
Average	11	0.362	0.312 (-13.7%)	0.412 (+13.7%)

Table 3.4: Test set performance for language pair Spanish→English

French→English				
Test set	Systems	Best	Worst	Optimal
nc-test2007	7	0.376	0.294 (-21.9%)	0.423 (+12.4%)
test2007	7	0.376	0.294 (-21.9%)	0.423 (+12.4%)
newstest2008	14	0.314	0.278 (-11.4%)	0.366 (+16.7%)
test2008	17	0.391	0.312 (-20.1%)	0.434 (+11.1%)
newstest2009	15	0.357	0.279 (-21.9%)	0.399 (+11.5%)
newssyscombtest2010	30	0.353	0.301 (-14.7%)	0.405 (+14.8%)
newstest2011	24	0.361	0.317 (-12.3%)	0.427 (+18.2%)
newstest2012	15	0.353	0.315 (-10.7%)	0.409 (+16.0%)
newstest2013	19	0.371	0.313 (-15.7%)	0.436 (+17.4%)
Average	16	0.361	0.300 (-16.7%)	0.413 (+14.5%)

Table 3.5: Test set performance for language pair French→English

Hungarian→English				
Test set	Systems	Best	Worst	Optimal
newstest2009	3	0.251	0.233 (-6.85%)	0.280 (+11.7%)
Average	3	0.251	0.233 (-6.85%)	0.280 (+11.7%)

Table 3.6: Test set performance for language pair Hungarian→English

Any→English				
Test set	Systems	Best	Worst	Optimal
newssyscombtest2010	3	0.397	0.380 (-4.30%)	0.414 (+4.29%)
Average	3	0.397	0.380 (-4.30%)	0.414 (+4.29%)

Table 3.7: Test set performance for language pair Any→English

English→Czech				
Test set	Systems	Best	Worst	Optimal
nc-test2007	2	0.200	0.189 (-5.40%)	0.216 (+7.95%)
nc-test2008	6	0.211	0.128 (-39.5%)	0.248 (+17.1%)
newstest2008	6	0.186	0.108 (-41.6%)	0.219 (+18.0%)
newstest2009	5	0.203	0.171 (-15.6%)	0.242 (+19.3%)
newssyscombtest2010	19	0.228	0.177 (-22.7%)	0.273 (+19.5%)
newstest2011	14	0.232	0.174 (-24.9%)	0.281 (+21.1%)
newstest2012	13	0.215	0.167 (-22.3%)	0.261 (+21.6%)
newstest2013	14	0.231	0.182 (-21.2%)	0.298 (+29.2%)
Average	7	0.206	0.155 (-25.0%)	0.240 (+16.4%)

Table 3.8: Test set performance for language pair English→Czech

English→German				
Test set	Systems	Best	Worst	Optimal
nc-test2007	6	0.244	0.185 (-24.2%)	0.284 (+16.3%)
test2007	7	0.271	0.208 (-23.2%)	0.336 (+23.9%)
newstest2008	11	0.217	0.198 (-8.90%)	0.268 (+23.7%)
test2008	13	0.262	0.187 (-28.7%)	0.306 (+16.5%)
newstest2009	11	0.225	0.198 (-12.1%)	0.278 (+23.7%)
newssyscombtest2010	22	0.249	0.196 (-21.4%)	0.293 (+17.6%)
newstest2011	35	0.236	0.185 (-21.7%)	0.289 (+22.3%)
newstest2012	15	0.239	0.200 (-16.3%)	0.291 (+21.8%)
newstest2013	21	0.261	0.220 (-15.6%)	0.329 (+26.1%)
Average	15	0.245	0.197 (-19.1%)	0.297 (+21.3%)

Table 3.9: Test set performance for language pair English→German

English→Spanish				
Test set	Systems	Best	Worst	Optimal
nc-test2007	7	0.337	0.309 (-8.34%)	0.386 (+14.5%)
test2007	7	0.337	0.262 (-22.5%)	0.408 (+21.0%)
newstest2008	12	0.263	0.241 (-8.29%)	0.316 (+20.4%)
test2008	15	0.321	0.236 (-26.3%)	0.389 (+21.3%)
newstest2009	9	0.300	0.218 (-27.2%)	0.330 (+10.1%)
newssyscombtest2010	20	0.325	0.252 (-22.5%)	0.364 (+12.1%)
newstest2011	23	0.316	0.256 (-18.9%)	0.369 (+16.8%)
newstest2012	11	0.323	0.273 (-15.7%)	0.372 (+15.1%)
newstest2013	18	0.316	0.267 (-15.6%)	0.373 (+18.0%)
Average	13	0.315	0.257 (-18.4%)	0.368 (+16.6%)

Table 3.10: Test set performance for language pair English→Spanish

English→French				
Test set	Systems	Best	Worst	Optimal
nc-test2007	8	0.314	0.244 (-22.4%)	0.354 (+12.7%)
test2007	8	0.329	0.277 (-15.9%)	0.399 (+21.3%)
newstest2008	13	0.264	0.215 (-18.6%)	0.311 (+17.8%)
test2008	14	0.326	0.165 (-49.4%)	0.378 (+15.9%)
newstest2009	12	0.290	0.228 (-21.3%)	0.337 (+16.3%)
newssyscombtest2010	22	0.306	0.239 (-21.9%)	0.356 (+16.4%)
newstest2011	22	0.335	0.239 (-28.6%)	0.384 (+14.8%)
newstest2012	15	0.303	0.263 (-12.9%)	0.360 (+19.0%)
newstest2013	21	0.314	0.259 (-17.4%)	0.377 (+20.3%)
Average	15	0.309	0.237 (-23.2%)	0.362 (+17.2%)

Table 3.11: Test set performance for language pair English→French

English→Russian				
Test set	Systems	Best	Worst	Optimal
newstest2013	18	0.399	0.357 (-10.4%)	0.445 (+11.5%)
Average	18	0.399	0.357 (-10.4%)	0.445 (+11.5%)

Table 3.12: Test set performance for language pair English→Russian

Russian→English				
Test set	Systems	Best	Worst	Optimal
newstest2013	23	0.346	0.262 (-24.3%)	0.409 (+18.2%)
Average	23	0.346	0.262 (-24.3%)	0.409 (+18.2%)

Table 3.13: Test set performance for language pair Russian→English

Development over Time

We now shift focus again and investigate how performance gains by oracle-based sentence selection have evolved over time.

Figure 3.9 depicts the evolution of average performance gains attained by optimal sentence selection for language pairs translating into English and involving Czech, German, Spanish, and French. We can clearly see that our approach achieves improvements independent of the language pair under investigation. Performance gains range from +5% up to +23% and seem to stabilise around +15% in 2010.

Figure 3.10 provides the same information for all language pairs translating from English into the aforementioned languages. Again, we observe consistent performance gains, ranging from around +7% and peaking at around +23%. The general level of potential improvement seems higher than in the previous graph. This may indicate that the computation of translations from English is more difficult still and results in lower $\text{Meteor}_{\text{AVG}}$ scores for which it is then easier to achieve bigger performance gains. This also supports the assumption that the lack of linguistic resources such as, e.g., synonymy knowledge bases similar to WordNet has a negative impact on the overall performance and utility of the Meteor metric.

In summary, the results from our sentence selection experiments clearly support our working hypothesis that the method is applicable irrespective of the creation date of the individual candidate translations.

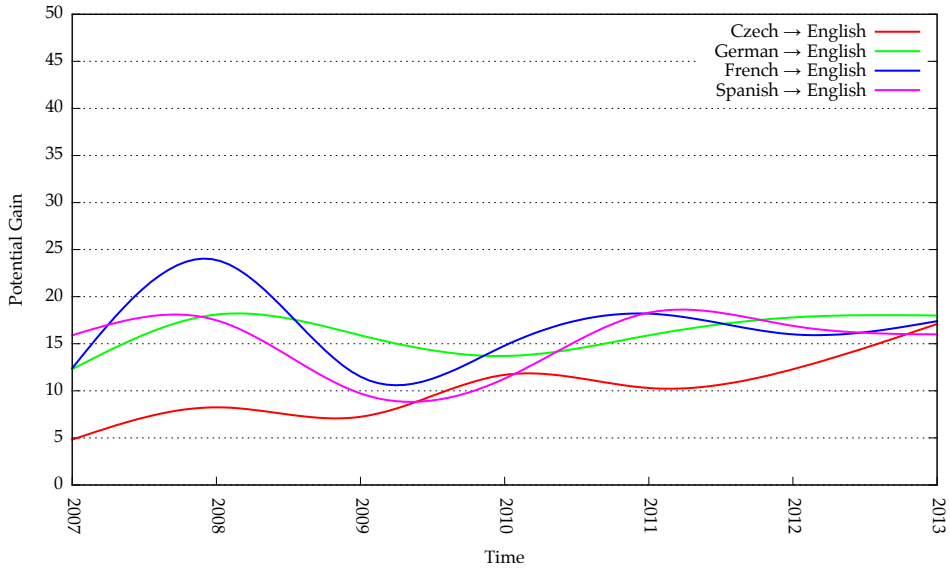


Figure 3.9: Average performance gain for sentence selection performance for translation into English over time, measured from 2007–2013.

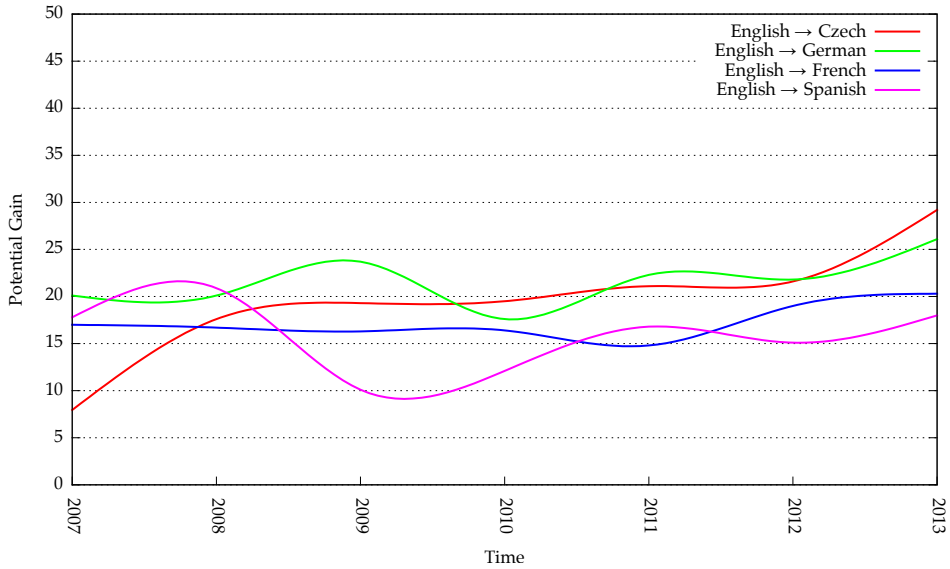


Figure 3.10: Average performance gain for sentence selection performance for translation from English over time, measured from 2007–2013.

Observations

Before concluding this chapter with a summary of our findings we want to briefly present some interesting observations from our experiments. These are depicted in Figure 3.11 which provides more details.

WMT 2007: German→English

This language pair shows that even *very bad* candidate systems may be beneficial for sentence selection approaches and very well can contain segment translations which can outperform the single-best baseline translation. Note how the *worst-2* setting nearly reaches the single-best Meteor score for this language pair. Adding the third worst candidate system outperforms this baseline score. This provides empirical evidence that even bad translation systems do in fact contain "nuggets" to mine for. This also confirms our research Hypothesis 3.

WMT 2008: English→German

This language pair shows that large performance gains can be achieved by implementing a simple sentence selection approach. In our experiments, we observed an overall improvement of +23.7% which turned a somewhat modest baseline score of 0.217 points $\text{Meteor}_{\text{AVG}}$ into a much better combined score of 0.268.

WMT 2009: English→Hungarian

For this language pair both participating systems, namely *morpho* and *uedin* perform on the same level of quality, around 0.17 $\text{Meteor}_{\text{AVG}}$ score. Still, the combination of both systems can outperform the individual candidate systems, achieving an improvement of +14.4%.

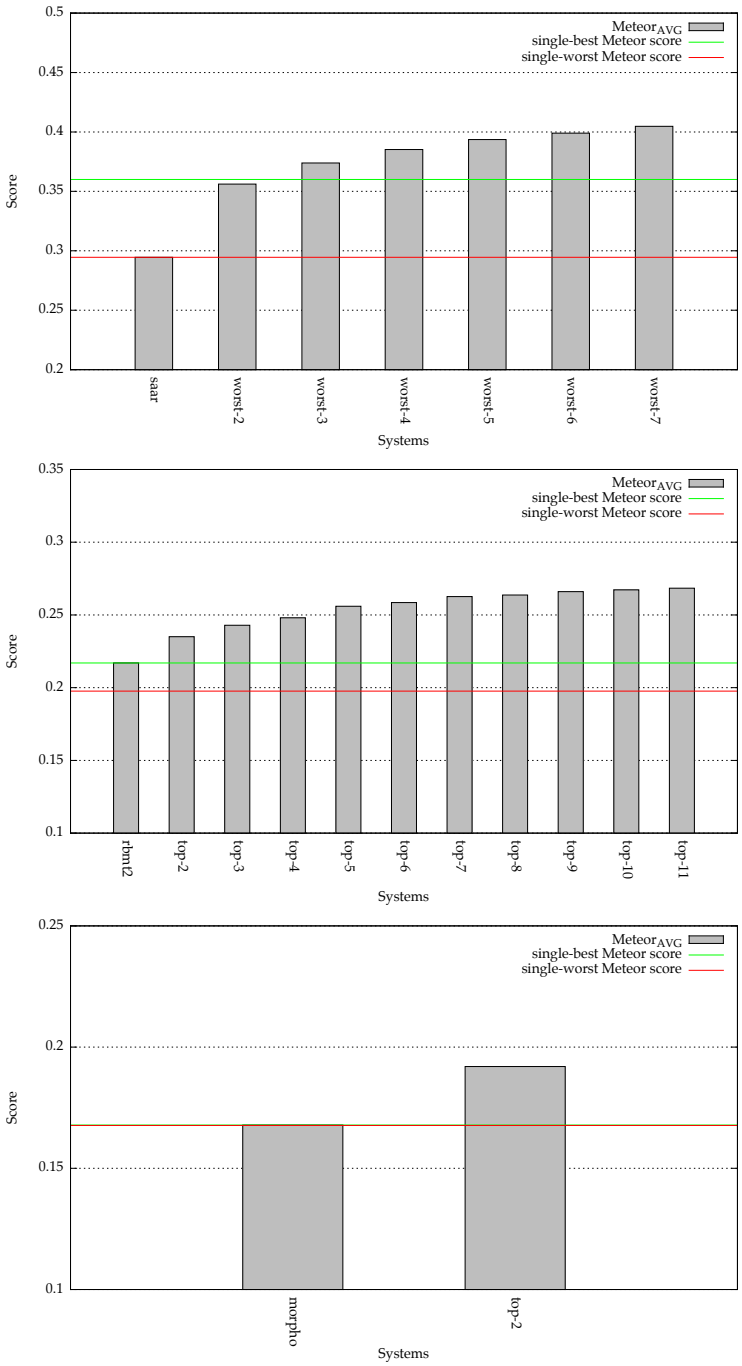
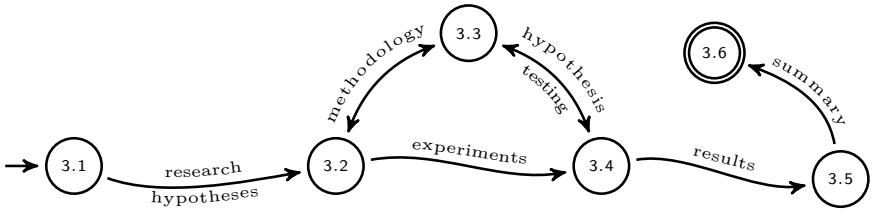


Figure 3.11: *Top:* combination of two very bad systems for German→English from WMT 2007 nearly outperforms the single-best baseline translation. *Middle:* combination for English→German from WMT 2008 yields a massive +23.7% performance gain. *Bottom:* combination of two equal systems for English→Hungarian from WMT 2009 results in a +14.4% increase in quality.

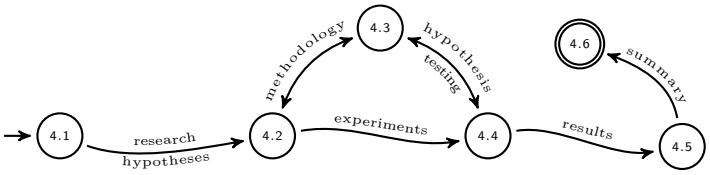


3.6 Chapter Summary

In this chapter we have investigated whether sentence selection can be used as a method for system combination in the context of machine translation of written text. For this, we have simulated optimal sentence selection on data sets taken from the yearly Workshop on Statistical Machine Translation (WMT) and the ML4HMT workshop series. We have defined three research hypotheses and then verified them one by one.

Sentence selection has proven to be an effective technique which can outperform the single-best translation baseline for a given set of candidate translations. This holds irrespective of external factors such as, e.g., language pair under investigation, quality of the individual candidate translations, or the underlying technological paradigms of the candidate systems. This implies that it is worthwhile pursuing experiments with sentence selection in *real world* application scenarios. By contrast to the research conducted in this chapter, such application needs proper classification between candidate systems which has to be modelled and learnt from training data.

We describe a method for training such classification systems for use with a sentence selection approach for system combination in the next chapter.



4 Joint, Comparison-Based Binarisation of Features

"The best way to predict the future is to invent it."

– **Alan Kay**: in 1971, at the Palo Alto Research Center (PARC).

"The aim of science is not to open the door to infinite wisdom, but to set a limit to infinite error."

– **Bertold Brecht**: *Life of Galilei*, 1939.

4.1 Introduction

In the previous chapter, we have described the results from our large-scale oracle experiments on WMT data. These results clearly indicate that system combination of MT output can be implemented using sentence selection as the methodological paradigm. In this chapter, we shift focus and investigate how such a selection approach can be best modelled using machine learning techniques. More specifically, we define a novel type of feature vectors which can be used with binary classification, e.g., with support vector machines (SVM) as introduced by [Vapnik, 1995]. By contrast to models which store feature values for single systems X only, we propose a new kind of joint feature vector which encodes feature values extracted from two systems A, B.

This technique allows us to explicitly model system comparison on the level of feature vectors. Intuitively, it seems plausible that the availability of all feature values for both systems under investigation should improve accuracy or performance of the resulting classification model. We carefully scrutinise this assumption in the remainder of this chapter.

Problem Statement

Let us first define the specific problem we are investigating:

> *Given single feature vectors and corresponding joint feature vectors on the same data. Can we observe improvements in terms of faster training time or better accuracy when using joint feature vectors?*

The problem of binary classification has seen extensive research over the past two decades. Hence, there exists a plethora of powerful machine learning methods which can be applied to solve such problems. The introduction of joint feature vectors has been inspired by the idea that, in order to properly compare two systems based on their individual feature values, any machine learning tool should have access to the joint set of feature values from both systems. This allows to use two target classes, +1 for "*system A is better than system B*" and −1 for the opposite event, and can theoretically be used with *any* machine learning approach which is based on feature vectors.

This chapter is structured as follows: in Section 4.2 we state our research hypotheses. Second, we describe the methodology (Section 4.3) we have used in our experiments which are discussed afterwards (Section 4.4). Finally, we present results (Section 4.5) before ending with a summary of our findings and a conclusion in Section 4.6.

4.2 Research Hypotheses

We want to verify that the application of joint feature vectors results in a higher prediction accuracy or faster training time when used in combination with state-of-the-art machine learning approaches. As a first step, we have to investigate whether such joint feature vectors can outperform and hence have an advantage over single feature vectors which have typically been used so far. An example is their application in the field of quality estimation. Here, each of the candidate systems is modelled using a set of single feature vectors which are then used to learn a classification model which estimates, usually by regression techniques, the overall quality of the given system. In our view, such methods waste potential in so far as they ignore *"local"*, i.e., sentence-level differences which could be observed by pairwise comparison of the systems' feature values. While the resulting regression score can well be an effective estimate of the system's quality, it seems plausible that, by making the individual feature values from two systems available inside one *joint* feature vector, a more fine-grained distinction among candidate systems can be achieved.

Thus, our first research hypothesis in this chapter thus is as follows:

Hypothesis 4. *Explicit modelling of pairwise system comparison using what we call joint feature vectors can outperform single feature vectors in terms of both resulting prediction accuracy and faster training times for large data sets.*

Second, we have further refined our notion of joint feature vectors by performing a *binarisation* on outcome of the comparison of the individual feature values for the two systems under investigation. We describe the exact definition of such feature vectors in the upcoming section. In a nutshell, however, the concept is easily explained like this: Given two sets X, Y of

feature values for a pair of systems A, B, we compute the joint, binarised feature vector by computing the binary result of all comparisons between "compatible" feature values. This means we store 1 in case the feature value X_i for system A is better than the corresponding feature value Y_i from system B, 0 (or -1) otherwise. A nice side effect of this method is that we usually have quite a good understanding of the feature spaces for our feature values (as we have defined those features before, we *should* know which score or value is supposed to be *"better"*) and can hence easily implement an appropriate comparison operator for each of these.

It seems reasonable to assume that joint, binarised feature vectors can outperform their joint counterparts. For joint feature vectors, the machine learning approach which is used has to *guess* which of the feature values correspond to each other and actually describe the same property of the two translations. While it is not impossible to learn such relationships from enough training data, we believe that it is better to perform comparison-based binarisation instead while the relationship between X_i and Y_i is still known. Thus, our second working hypothesis is the following:

Hypothesis 5. *Binarisation of the comparison results obtained by comparing the individual feature values from two systems A, B can outperform the corresponding joint feature vectors.*

We verify both hypotheses in the remainder of this chapter. In order to do so, we simulate perfect feature values (similar to our oracle approach in Chapter 3) and train classification models based on 1) single feature vectors with four features per vector, 2) joint feature vectors with eight features per vector, and finally 3) joint, binarised feature vectors which again have only four features per vector due to the comparison of feature values from the two

systems. To make the problem increasingly harder, we add random Gaussian noise to the feature vectors, effectively increasing the number of features per vector up to a total of 10,000 noisy values per vector.

4.3 Methodology

Definitions

First, we have to properly define the subject of investigation. We define the notions of 1) single, 2) joint, and 3) joint, comparison-based binarisation of feature vectors. Afterwards we discuss how instances of such feature can be created from translation output obtained from a set of different candidate systems.

Definition 4 (Single Feature Vectors)**.** *A feature vector $X \in \mathbb{R}^n$ which does only contain feature values $f_i(A), 1 \leq i \leq n$ which have been extracted from a single candidate translation A, not considering any other translation B, is called* single feature vector.

Formally, we define $X_{single}(A) \in \mathbb{R}^n$ as follows:

$$X_{single}(A) \quad \stackrel{\text{def}}{=} \quad \begin{pmatrix} f_1(A) \\ f_2(A) \\ \vdots \\ f_n(A) \end{pmatrix} \tag{4.1}$$

By contrast, *joint feature vectors* (as the name implies) are composed of two sets of feature values which have been extracted from two candidate systems A, B. Such feature vectors $X \in \mathbb{R}^{2n}$ first contain the individual feature values from system A, followed by the individual feature values from system B. Effectively, we are "concatenating" two single feature vectors $X_{single}(A)$ and $X_{single}(B)$ when constructing a vector $X_{joint}(A, B)$.

Definition 5 (Joint Feature Vectors)**.** *A feature vector $X \in \mathbb{R}^{2n}$ which is comprised of feature values $f_i(A), 1 \leq i \leq n$ followed by feature values $f_i(B), 1 \leq i \leq n$ which have been extracted from two candidate translations A, B, is called* joint feature vector.

Formally, we define $X_{joint}(A, B) \in \mathbb{R}^{2n}$ like this:

$$X_{joint}(A, B) \stackrel{\text{def}}{=} \begin{pmatrix} X_{single}(A) \\ X_{single}(B) \end{pmatrix} \tag{4.2}$$

Or, written verbosely:

$$X_{joint}(A, B) \stackrel{\text{def}}{=} \begin{pmatrix} f_1(A) \\ \vdots \\ f_n(A) \\ f_1(B) \\ \vdots \\ f_n(B) \end{pmatrix} \tag{4.3}$$

The binarisation step of a so-called *joint*, *binarised feature vector* computes the comparison results for all compatible feature values from two candidate translations A, B, i.e., all values $\text{cmp}(f_i(A), f_i(B)), 1 \leq i \leq m$. The outcome of any such comparison operator cmp can be in range $\{-1, 0, 1\}$. While this, strictly spoken, implies that our resulting feature vectors contain *ternary* feature values, we stick to the original *binary* name as it was used in the early phases of this dissertation work when $\text{cmp}(x, y)$ was implemented as $x \geq y$, hence only producing binary results.

Definition 6 (Joint, Comparison-Based Binarisation of Features)**.** *A feature vector $X \in \mathbb{R}^n$ which is comprised of feature values which compare individual feature values $f_i(A)$ and $f_i(B), 1 \leq i \leq n$ which have been extracted from two candidate translations A, B and store a binary (or ternary) result as corresponding feature value, is called* joint, binarised feature vector.

Formally, we define $X_{binarised}(A, B) \in \mathbb{R}^n$ *like this:*

$$X_{binarised}(A, B) \quad \stackrel{\text{def}}{=} \quad \begin{pmatrix} cmp(f_1(A), f_1(B)) \\ cmp(f_2(A), f_2(B)) \\ \vdots \\ cmp(f_n(A), f_n(B)) \end{pmatrix} \tag{4.4}$$

The difference between binary and ternary feature values

As we have seen, it is possible to define the comparison operator $\text{cmp}(x, y)$ which is used to produce the "binarised" feature values for $X_{binarised}(A, B)$ s.t. it either has a binary or a ternary output range. The need for a ternary solution comes from the fact that the comparison of feature values can in fact result in a draw, i.e., $f_i(A) = f_i(B)$. This is especially true for integer or float scores, or any other format for which numerical equality is defined. For many machine learning techniques, however, it is beneficial (and hence recommended) to scale individual feature values to range $[0, 1]$. This may result in a more efficient training process which is, of course, a desirable property for any machine learning problem.

In our experiments with joint, comparison-based binarisation of feature vectors, we have found that using comparison operators with ternary range works nicely. This holds for both rescaling the resulting feature values to range $[-1, 1]$, using target values of $\{-1, 0, 1\}$ and also for rescaling to range $[0, 1]$, using target values of $\{0, 0.5, 1\}$. For machine learning algorithms which strictly require a binary feature value range, i.e., either $\{-1, 1\}$ or $\{0, 1\}$, there are three conversion strategies for feature value comparison:

1. Use $x \geq y$ as comparison operator $\text{cmp}(x, y)$. This makes sense as the comparison of two equal systems can be seen in a reflexive way, meaning that both systems are, at the same time, better than the other. It

is questionable, though, if the resulting feature vectors can contribute much to the desired classification model. Their "semantic content" just does not help to find out which of the two given systems is deemed the better one;

2. A second option is to remove any feature vectors which would require a ternary output range for any of their feature values from the training data. While this seems a reasonable choice, it can result in a *massive* reduction of feature vector instances in the training data and, thus, a worse prediction accuracy of the resulting classification model;
3. Some machine learning approaches allow for sparse feature vectors. These may omit feature values for a subset of the defined features. In our scenario, we would omit any feature value for which $f_i(A) = f_i(B)$ and then normalise to the desired target range, either $\{-1, 1\}$ or $\{0, 1\}$.

After having defined the fundamental concepts of this chapter, we now describe the experiments we have conducted in order to verify the research hypotheses stated in Section 4.2.

4.4 Experiments

In order to investigate the performance of the three types of feature vectors defined in the previous section, we conduct a series of experiments which are described below. In these experiments, we use perfect feature values, obtained by an oracle and available in all feature vector types, and then train classification models with various machine learning techniques. We use k-fold cross validation to compute the prediction accuracy of the resulting models. Based on these data, we can find out whether our working hypotheses actually hold and verify or falsify them.

Data

We work on (anonymised) data taken from the NIST OpenMT12 shared task. The data set consists of 127 sentence translations obtained from a total of 15 individual translation systems. The text domain was newswire text which was translated from Chinese into English. Using the supplied reference text, we compute four "perfect" feature values for each of the translations:

1. Meteor [Denkowski and Lavie, 2011] computed on the segment level;
2. Meteor computed on the corpus level. By contrast to the Meteor_{AVG} scores reported in the previous chapter, where runtime was an issue s.t. we wanted to prevent lengthy re-computation of the Meteor scores, we use the "full" corpus-level scores reported by the Meteor tool.
3. NIST [Doddington, 2002] computed on the corpus level; and
4. BLEU [Papineni et al., 2002] also computed on corpus level.

In our research on hybrid machine translation and evaluation of machine translation output, we have found that Meteor often has the best correlation with human judgment. Furthermore, it is the only metric which can generate meaningful segment scores.[1] For these reasons, the preceding enumeration of features also determines our preferred ordering of these features. When comparing two systems based on these, we first compare feature 1, Meteor on the segment level. Only if both systems score the same we consider feature 2, Meteor on the corpus level. This recursive definition of our decision function continues down to the level of BLEU scores on the corpus level. We will provide a proper definition in the next chapter where we define our hybrid combination framework.

[1] There exist other metrics which have reliable sentence scores, e.g., s-BLEU or TER. These could also be used instead of or in combination with Meteor.

	Metric Scores				
System	Meteor_{system}	Meteor_{best}	Meteor_{worst}	NIST	BLEU
#1	0.2994	1.0000	0.1218	7.9417	0.3187
#2	0.2405	1.0000	0.0000	5.3019	0.1937
#3	0.2418	0.9143	0.0726	6.1327	0.1641
#4	0.3154	1.0000	0.1288	8.3071	0.3476
#5	0.3047	1.0000	0.1553	8.0122	0.3238
#6	0.2846	0.9280	0.1282	7.6076	0.2925
#7	0.2700	0.5465	0.0460	6.8932	0.2539
#8	0.2773	0.8480	0.0987	7.2073	0.2502
#9	0.2935	1.0000	0.1103	7.6039	0.2885
#10	0.2750	1.0000	0.0825	7.1198	0.2392
#11	0.3011	1.0000	0.1060	7.7787	0.3073
#12	0.2804	1.0000	0.0958	7.0356	0.2754
#13	0.2871	1.0000	0.1194	7.4073	0.2592
#14	0.2994	1.0000	0.1115	7.7210	0.2881
#15	0.3002	1.0000	0.1090	7.8369	0.3034
Average	0.2847	0.9491	0.0991	7.3271	0.2737

Table 4.1: Data set statistics for our feature vector experiments

Table 4.1 gives an overview on the individual quality of the 15 candidate systems from our data set. We present the overall Meteor_{system} score as well as both the best and the worst individual segment scores, Meteor_{best} and Meteor_{worst}, respectively. Following the definition of our feature values above, we also list NIST and BLEU scores for each system.

Single versus joint feature vectors

It is difficult to use single feature vectors to implement pairwise comparison of candidate translations A, B. For classification models in machine learning, multi-class problems, still represent a harder problem than classification problems with only two target classes +1 and −1. Also, there exist more machine learning tools to solve binary classification problems which suggests

that it makes sense to use such a method to estimate the performance level of single feature vectors.

In our experiments, we implement such a two class scenario by creating a total of $\frac{N(N-1)}{2} = 105$ sub data sets for our $N = 15$ candidate systems. Each of these data sets represents a pairwise comparison of two systems A, B. Our target class +1 denotes that, according to the evidence in the feature values, system A is better than system B. Conversely, system B would then get a target class of -1 as, based on the evidence in the feature values, it performs worse than system A.

For each of the 105 sub problems, we compute classification models (which differ in the amount of noise, the kernel type, and machine learning approach used) and evaluate their prediction accuracy using k-fold cross validation. We compute the average accuracy $accuracy_{AVG} \stackrel{\text{def}}{=} \frac{1}{105}\Sigma_{i=1}^{105} accuracy_i$ and use this score to compare the performance of single feature vector models against models based on joint feature vectors. We also record the time it takes to train the classification models, measuring how long the training process for all noise levels and cross-validation fold settings takes.

We use **scikit-learn** [Pedregosa et al., 2011], a Python-based framework which provides access to several, state-of-the-art machine learning tools, to implement our experiments. Specifically, we use the wrappers scikit-learn provides to access 1) **libSVM** [Chang and Lin, 2011] and 2) a more efficient implementation of linear classifiers named **liblinear** [Fan et al., 2008]. The former toolkit implements SVM models and is available as `svm.SVC` (we only use the classification variant), the latter is an optimised version of libSVM, implemented as `svm.LinearSVC`. Next to the core machine learning models, we also make use of scikit-learn's implementation of k-fold cross validation and stratified folding.

Joint versus joint, binarised feature vectors

By their very design, joint feature vectors are capable of explicitly modelling the comparison of two candidate systems—this is exactly the task we have defined them for. In order to compare joint feature vectors to their binarised counterparts, we compute two data sets:

1. first, one containing joint feature vectors for a total of $N * K = 13,335$ feature vector instances which correspond to all feature vectors for $N = 105$ system comparisons and $k = 127$ sentences for each system. Each of the feature vectors contains exactly $2 \times 4 = 8$ feature values plus an increasing number of noisy features. We provide more details on the noise levels below; and
2. a second data set which contains a total $N * K = 13,335$ joint, binarised feature vector instances, again corresponding to all feature vectors for all system comparisons and sentences for which we have translations to consider. Similar to the first data set, we also add increasing levels of noise to this data set.

An immediate observation can be made in contrast to the methodology applied for our experimental setup concerning classification models trained on single feature vector instances: instead of training the classifier on a total of $N = 105$ individual classification models with a maximum of $2 * K = 254$ feature vector instances (remember that each pair of systems yields two feature vectors per sentence for single feature vectors), we can leverage the full amount of $N * K = 13,335$ feature vector instances when working with (both types of) joint feature vectors. This represents an increase of factor $\times 105$ in terms of training data—which is a *massive* increase that may help to obtain more accurate prediction models. *"There is no data like more data!"*

The impact of noise

Training a machine learning classification model on training instances which consist of only four or eight feature values may not be a sufficiently hard challenge for either of the three feature vector types. This especially holds true considering that we use perfect feature values which are directly related to the respective target class. Hence we make the problem more difficult by adding *noisy feature values*. These are randomly drawn from a discrete uniform distribution in the closed set $\{-1, 0, 1\}$ (or $\{0, 1\}$ in case of strictly binary feature values). We add increasingly larger amounts of noise to our feature vectors and then train new classification models on the augmented set of training instances.

In total, we test ten different setups with the following noise levels:

- **noise** = 0: the baseline case with no additional noise. In theory, all feature vector types should perform best for this setup as there is no random noise which can interfere with the estimation of the classifier;
- **noise** = 10: a modest amount of 10 noisy features (factor ×2.5);
- **noise** = 20: a modest amount of 20 noisy features (factor ×5);
- **noise** = 50: a medium amount of 50 noisy features (factor ×12.5);
- **noise** = 100: a medium amount of 100 noisy features (factor ×25);
- **noise** = 200: a medium amount of 200 noisy features (factor ×50);
- **noise** = 500: a large amount of 500 noisy features (factor ×125);
- **noise** = 1,000: a large amount of 1,000 noisy features (factor ×250);
- **noise** = 5,000: a huge amount of 5,000 noisy features (factor ×1,250);
- **noise** = 10,000: a huge amount of 10,000 noisy features (factor ×2,500). This amount of noise does have a huge impact both on training time required for classifier estimation as well as the resulting accuracy.

4.5 Results

Single versus joint feature vectors: Accuracy

Table 4.2 shows the accuracy we observe for classification models trained using single feature vectors. Noise level ranges from 0 to 100 only as the small amount of training instances, $2 \times 127 = 254$ for single vectors and only 127 in case of the corresponding joint feature vectors, does not allow the resulting models to achieve more than chance prediction accuracy for larger numbers of noisy features. We apply k-fold cross validation with $k \in \{3, 5, 10\}$.

The addition of noisy features degrades the accuracy of the classifier. Without noise, we observe an average $\text{accuracy}_{AVG} = 0.69$ for $k = 3$ folds. For both $k = 5$ and $k = 10$ folds, this value is slightly better at 0.70. Each iteration then increases the amount of noise which is added to the feature vector, resulting in a decrease of prediction accuracy. For noise level 100 and $k = 5$ folds, accuracy reaches the minimum value at 0.56.

Table 4.3 gives the corresponding accuracy values we measure for models which have been built using joint feature vectors. As a general observation we can state that such classifiers achieve a higher maximum quality, 0.76 instead of 0.69. Otherwise, scores and their degradation caused by the addition of increasing levels of noise are comparable to the case of single feature vectors. Models based on joint feature vectors seem to have a slight edge on their single vector counterparts, but are not much better for noisy features.

Figure 4.1 plots prediction accuracy against the amount of noisy features. The (*Top*) graph refers to single feature vectors. The (*Middle*) graph shows results for joint feature vectors. The (*Bottom*) graph contrasts the two best models from both methods. The joint model on $k = 3$ folds outperforms the best single model until noise level 20. This supports Hypothesis 4.

Single versus joint feature vectors: Training time

Table 4.4 lists the training time for each of the single vector models we train. Without noise, only using the set of $2 \times 127 = 254$ training vector instances, the estimation of the resulting classification model is complete in a matter of seconds. For $k = 3$ folds, we measure roughly 1.8 seconds. Of course, this increases for larger folds as there is more computation required for these. We observe around 3.9 seconds for $k = 5$ and 7.8 seconds for $k = 10$.

The introduction of additional noisy features slows down the training process of the classifier. For noise level 10, training time rises by a factor of $\times 7$ to around 14.1 seconds. Again, larger folds take longer to complete. We observe an increase by factor $\times 8$ to 30.9 seconds for five folds. For $k = 10$ we record a total of 70.1 seconds, which is a jump by a factor of $\times 9$. These patterns continue for each of the iterations of our experiment.

Table 4.5 shows how training time evolves for classification models based on joint feature vectors. Again, training time without noise is a matter of seconds, 1.2 seconds for $k = 3$ folds. 2.3 and 4 seconds for $k = 5$ and $k = 10$ folds, respectively. Joint feature vectors without noise outperform their single vector counterparts (again supporting Hypothesis 4). The addition of noisy features further adds to this observation. For $k = 3$ folds, *every* of the measured training times for joint feature vectors, from noise level 0–100, takes less time than the training time for our single vector model with noise level 10: around 5.2 seconds versus around 14.1 seconds for the classifier using single feature vectors. The same holds for $k = 5$ and $k = 10$ folds.

Figure 4.2 visualises these numbers. The (*Top*) graph plots training time of single vector models against the amount of noisy features. The (*Middle*) graph provides the same information for joint feature vectors while, again, the (*Bottom*) graph contrasts the best models from both methods.

	Accuracy single vectors		
Noise level	3 folds	5 folds	10 folds
0	0.69 (± 0.08)	0.70 (± 0.08)	0.70 (± 0.08)
10	0.67 (± 0.10)	0.68 (± 0.09)	0.69 (± 0.08)
20	0.66 (± 0.10)	0.67 (± 0.10)	0.68 (± 0.09)
50	0.60 (± 0.07)	0.62 (± 0.08)	0.63 (± 0.08)
100	0.57 (± 0.08)	0.56 (± 0.07)	0.57 (± 0.07)

Table 4.2: Accuracy of single feature vectors

	Accuracy joint vectors		
Noise level	3 folds	5 folds	10 folds
0	0.73 (± 0.08)	0.75 (± 0.07)	0.76 (± 0.07)
10	0.72 (± 0.08)	0.74 (± 0.06)	0.74 (± 0.07)
20	0.66 (± 0.09)	0.70 (± 0.08)	0.71 (± 0.08)
50	0.59 (± 0.08)	0.60 (± 0.07)	0.62 (± 0.08)
100	0.58 (± 0.10)	0.58 (± 0.09)	0.57 (± 0.09)

Table 4.3: Accuracy of joint feature vectors

	Training time single vectors *[min:s:µs]*		
Noise level	3 folds	5 folds	10 folds
0	00:01.8544	00:03.9197	00:07.8316
10	00:14.1451	00:30.8937	01:10.0839
20	00:19.4627	00:43.8604	01:43.9383
50	00:44.2777	01:49.7188	04:33.0468
100	00:28.0203	01:41.4304	05:51.3853

Table 4.4: Training time of single feature vectors

	Training time joint vectors *[min:s:µs]*		
Noise level	3 folds	5 folds	10 folds
0	00:01.2411	00:02.2835	00:04.0302
10	00:02.5407	00:04.8859	00:10.9712
20	00:04.4051	00:08.9651	00:22.2281
50	00:05.2053	00:14.6410	00:42.7394
100	00:04.1371	00:09.1414	00:23.7568

Table 4.5: Training time of joint feature vectors

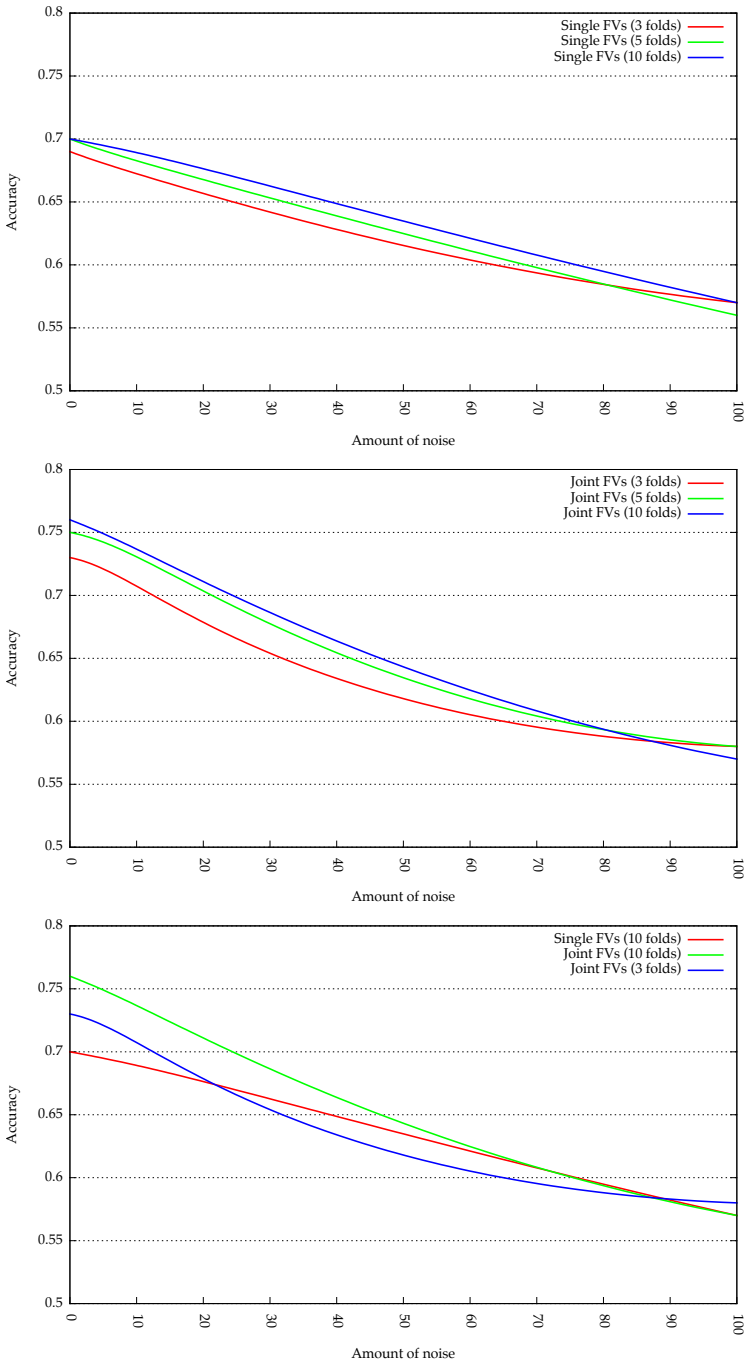


Figure 4.1: *Top:* accuracy for single feature vector classification models with increasing noise level. *Middle:* accuracy for joint feature vector classification models with increasing noise level. *Bottom:* comparison of the two most accurate models of both single and joint feature vector classification (for 10 folds). Note how the joint feature vectors model trained on 3 folds has a comparable accuracy, regardless of the decrease in folds.

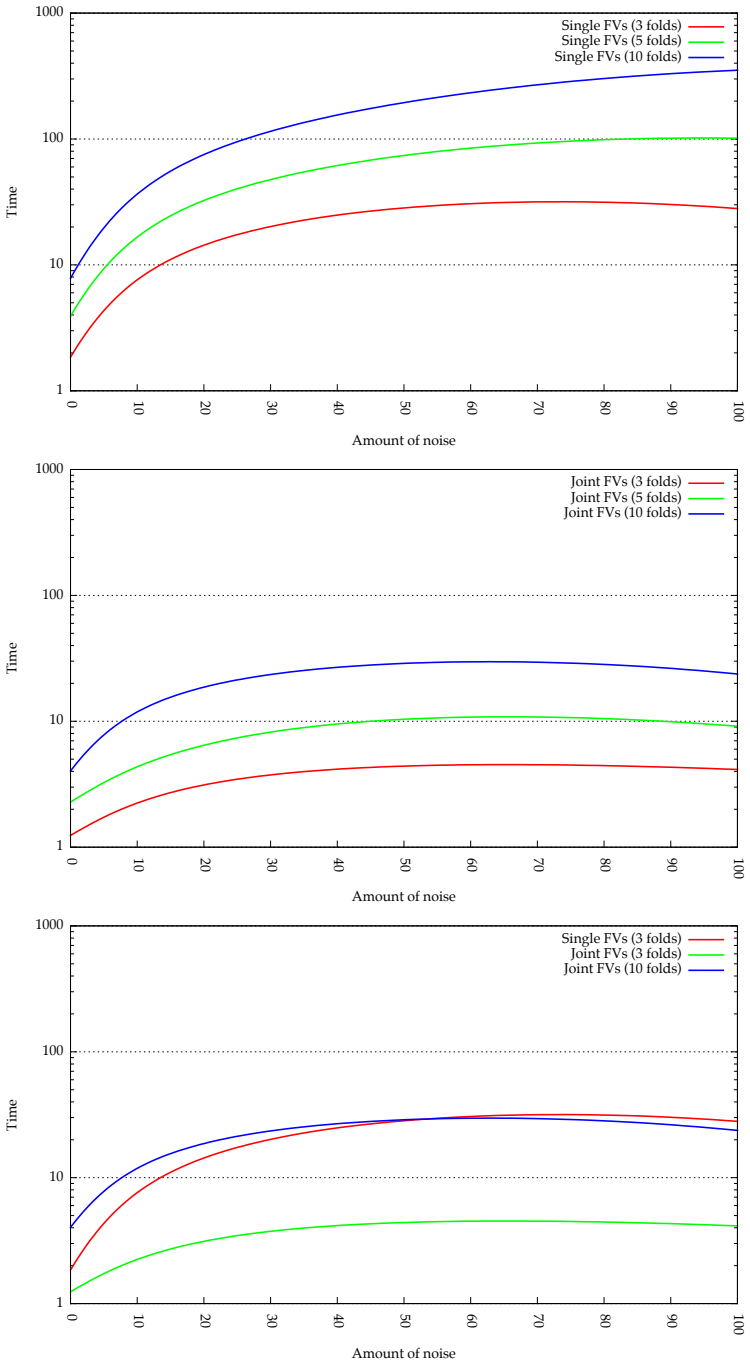


Figure 4.2: *Top:* training time for single feature vector classification models with increasing noise level. *Middle:* training time for joint feature vector classification models with increasing noise level. *Bottom:* comparison of the two fastest models of both single and joint feature vector classification (for 3 folds). Note how the joint feature vectors model trained on 10 folds has a comparable runtime, regardless of the increase in folds.

Joint versus joint, binarised feature vectors

We present results from our experiments comparing joint feature vectors to their binarised counterparts on the upcoming pages. The accuracy of binary classification models can be presented graphically using so-called *receiver operating characteristics* (ROC), or simply ROC curves. A ROC curve plots the fraction of *true positives* out of all positively labeled targets (this fraction is usually called TPR, *true positive rate*) against the fraction of *false positives* out of all negatively labeled targets (analogously, this is called FPR or *false positive rate*), at various threshold parameters which control the estimation process of the classification model under investigation.

A ROC curve displays FPR on the x-axis and plots TPR on the y-axis. The diagonal from $(0,0)$–$(1,1)$, i.e., from the lower left corner to the upper right corner of the graph, represents the so-called *line of no-discrimination*. This line represents the results of *random guessing*, i.e., a classification model which is practically useless. Any point above the diagonal represents a model with a "better than random" classification quality. Conversely, any point below the diagonal signals a model which performs very poorly, even worse than random. A perfect classification would be represented by point $(0,1)$. In essence: the further the plotted ROC curve is above the diagonal, the better are the underlying classification models.

Our experiments with joint and joint, binary feature vectors focus on the comparison of the influence of the comparison-based binarisation on the resulting classification models' prediction accuracy. We compute models for subsets of the full data set which contains $13,335$ feature vectors for $N = 105$ comparisons of 127 individual sentences. Our subset size ranges from $2,000$ to $12,000$. We use cross validation with $k = 5$ folds and add increasing amounts of noise to the feature vectors, similar to our methodology when

comparing single and joint feature vectors.

Classification models based on `svm.LinearSVC`

The first series of ROC curves (Figures 4.3–Figure 4.7) shows how prediction accuracy of joint feature vectors differs from the quality of the corresponding classification models trained on joint, binarised feature vectors. Graphs on the left side give an overview on the performance of joint feature vectors, for noise levels 0, 100, 1,000, and 10,000 from top to bottom. Likewise, the graphs on the right side show the same information for joint, binarised feature models. The plots also give the *Mean ROC area*, which is a "score", ranging from 0.0 to 1.0. Large mean ROC area values represent models with very good discrimination capabilities. The closer this value gets to 0.5, the less reliable the predictions of the corresponding model are.

Figure 4.3 presents results from classification models trained on a subset of 2,000 sentences. We use a linear model and apply `svm.LinearSVC` from scikit-learn for classifier estimation. The mean ROC area values for joint feature vectors range from 0.99, which essentially is near perfect prediction accuracy, down to 0.53, only minimally better than random guessing. Models trained using joint, binarised feature vectors show a better performance with mean ROC area values ranging from 0.99 to 0.73. While the latter value certainly is not stellar, it still is much better than the corresponding value from the model trained on joint feature vectors. It is also noteworthy that the introduction of increasingly more noisy features does not have as much of a negative effect on joint, binarised feature vectors as it has on their joint counterparts. Binarisation of feature comparison results seems to be beneficial, supporting Hypothesis 5. Other plots on pages 4.4–4.7 offer similar insights.

We describe the performance of a second, potentially more powerful SVM

implementation, namely `svm.SVC`, in combination with two different kernel types, "linear" and "rbf" (which stands for *radial basis function*), next.

Classification models based on `svm.SVC` with linear kernel

In principle, the combination of `svm.SVC` as implemented by **libSVM** with a linear kernel should yield results which are very similar to the experiments conducted with `svm.LinearSVC`. As previously mentioned, the latter is only a more efficient implementation of linear models, especially with thousands of feature values per feature vector. The improved runtime performance, of course, comes at a cost; this becomes clear when evaluating the ROC curves plotted in (Figures 4.8–Figure 4.12). Note that each and every of the plots achieves a higher prediction accuracy than the corresponding models trained using `svm.LinearSVC`. As expected, the increased complexity of the `svm.SVC` estimation results in greater runtime requirements. The choice among the two implementations is hence based on application requirements.

Figure 4.8 presents results from classifiers based on a subset of 2,000 sentences, similar to plots on page 98 but for the `svm.SVC` implementation. The mean ROC area values for joint feature vectors range from 0.99, which is near perfect prediction accuracy, down to 0.53, which is only slightly better than random guessing. Models trained using joint, binarised feature vectors show a better performance with mean ROC area values ranging from 0.99 to 0.83. The latter being +0.10 larger than the corresponding value depicted in Figure 4.3 (*Bottom*, *right*). Also note that joint, binarised feature vectors only drop for $k = 10{,}000$ noisy features, otherwise showing stellar accuracy values of 0.99, except for noise level 5,000 (not pictured) which achieves an accuracy of 0.91. Again, it seems that comparison-based binarisation better copes with noise.

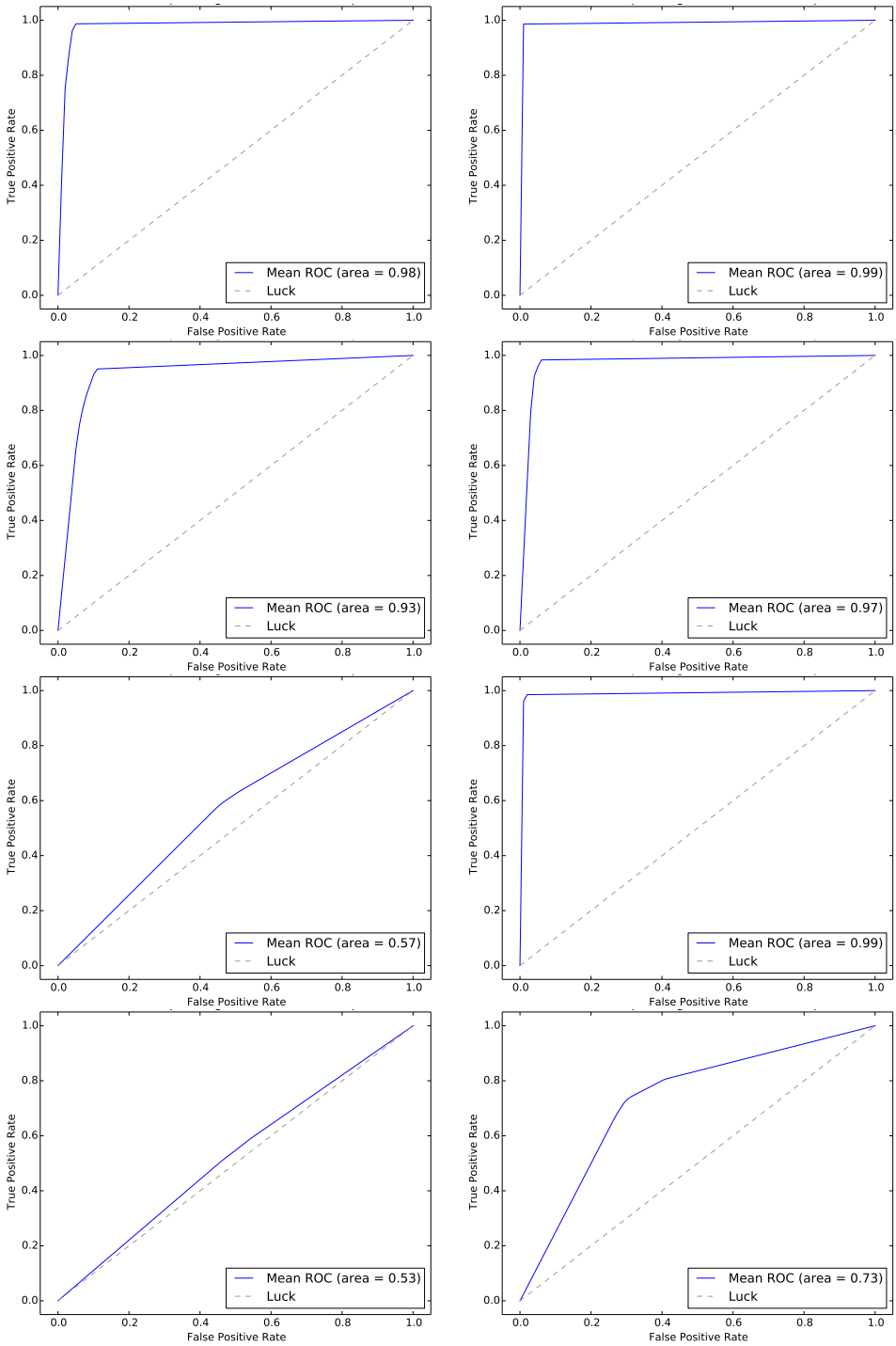


Figure 4.3: **Subset: 2,000, Kernel: linear, Type: LinearSVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

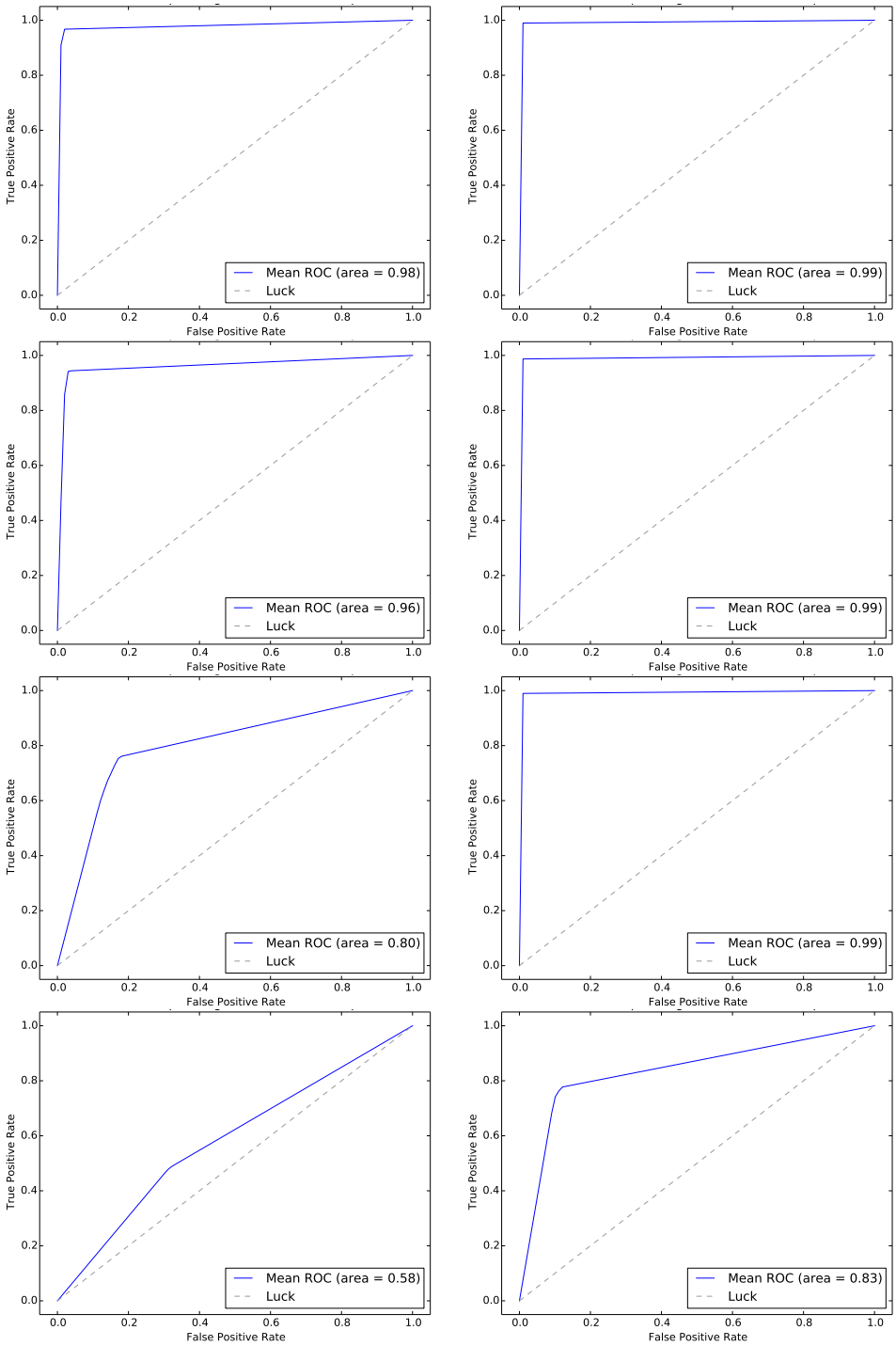


Figure 4.4: **Subset: 4,000, Kernel: linear, Type: LinearSVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

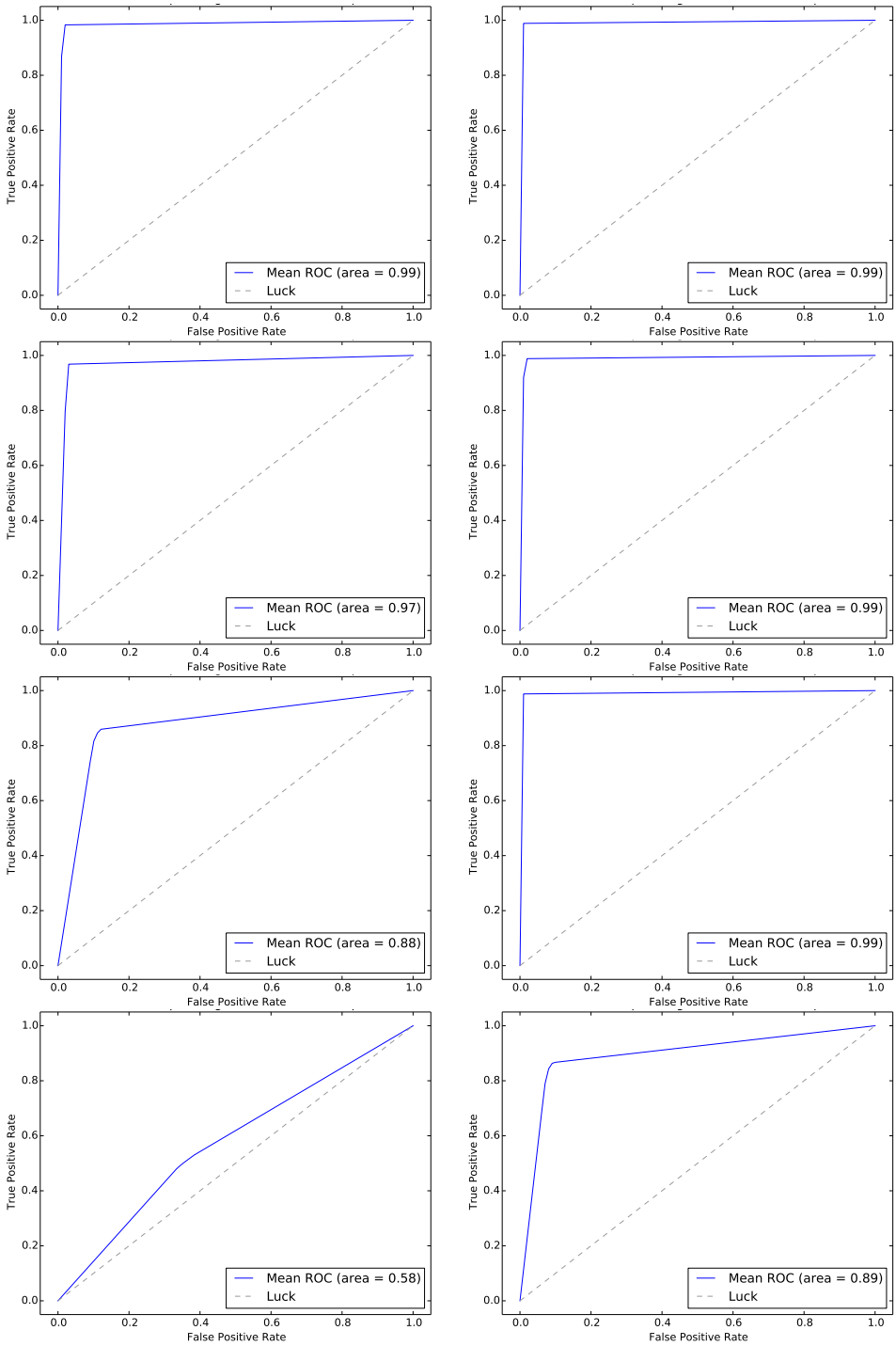


Figure 4.5: **Subset: 6,000, Kernel: linear, Type: LinearSVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

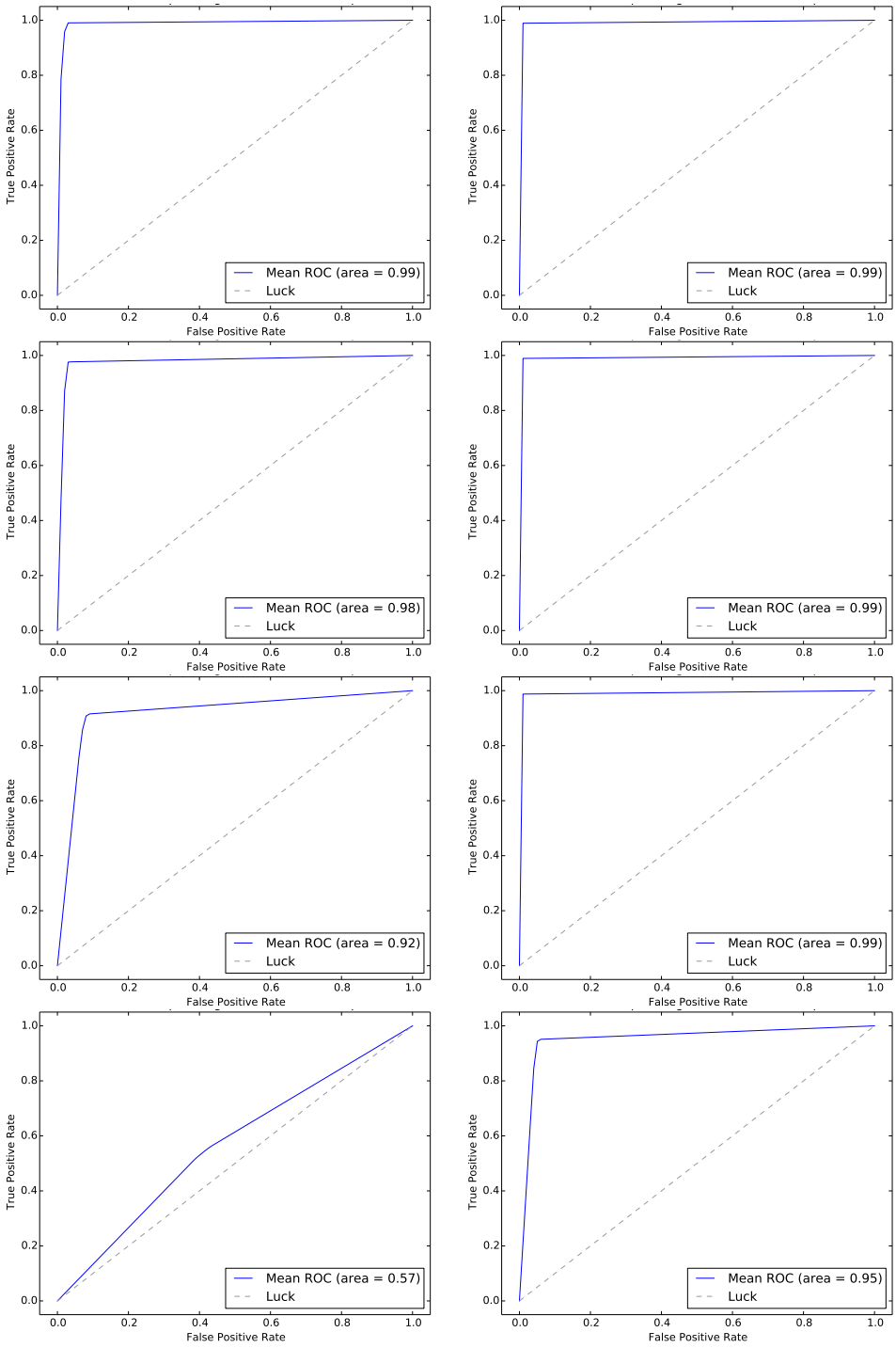


Figure 4.6: **Subset: 9,000, Kernel: linear, Type: LinearSVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

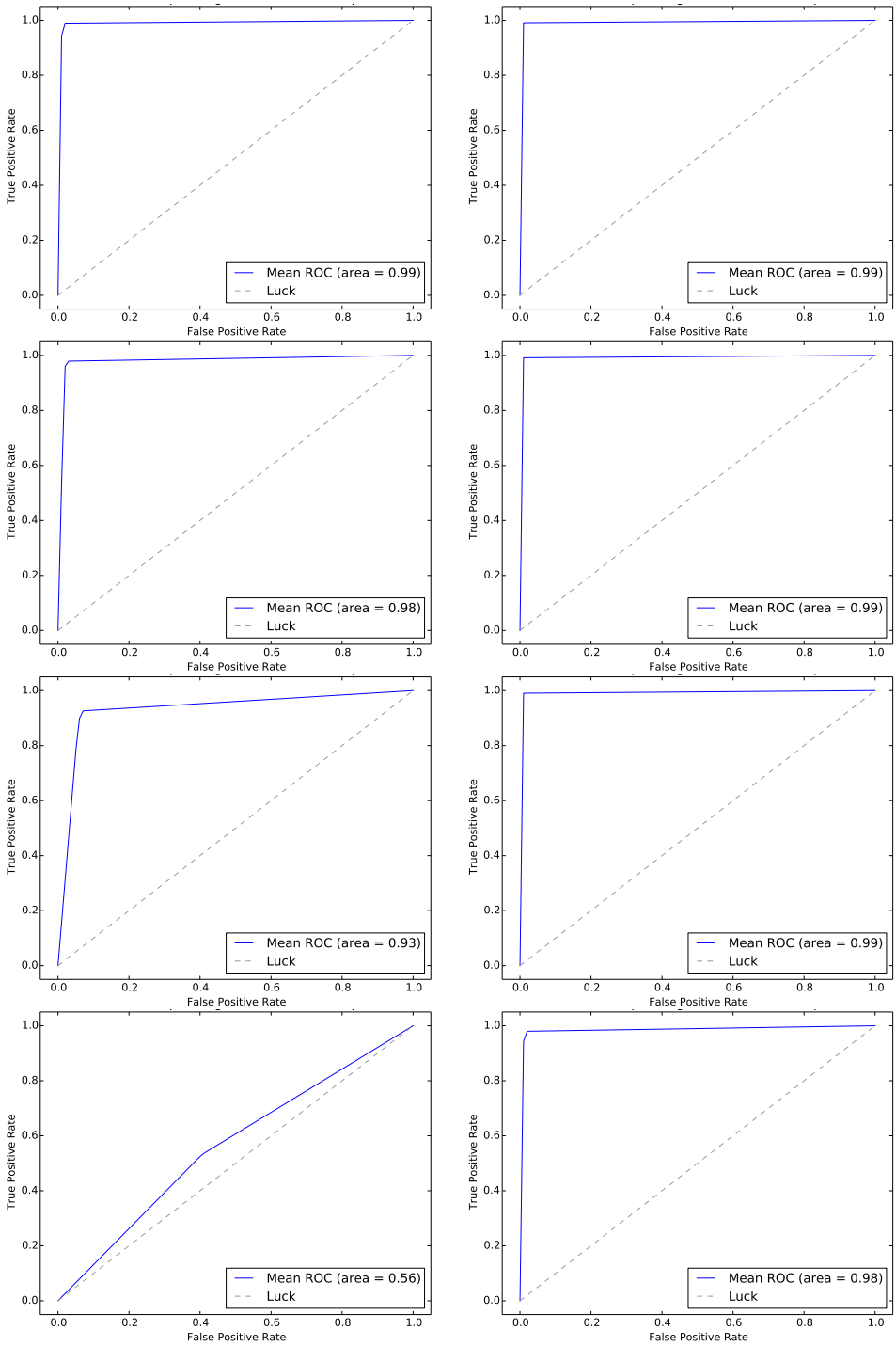


Figure 4.7: **Subset: 12,000, Kernel: linear, Type: LinearSVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

Similar to our previous findings, prediction accuracy improves relative to increasing subset (or training set) size. While joint vectors get slightly better, the binarised variants perform much better. Mean ROC area for Figure 4.12 (*Bottom right*) remains at 0.99, compared to a joint score of only 0.58, a much worse value.

Classification models based on `svm.SVC` with rbf kernel

Finally, we discuss results from our experiments with classification models trained using the rbf kernel. As mentioned before, the term *rbf* stands for *radial basis function*, i.e., the kernel function $e^{-\gamma\|x-x'\|^2}$. Parameter γ has to be greater than 0. The use of an rbf kernel allows the support vector machine to create a nonlinear classifier by applying the so-called *kernel trick*. The kernel function transforms feature values into a Hilbert space of infinite dimensions, potentially resulting in a binary classification model which is able to achieve better discrimination capabilities on the given training data. Estimation of such a classifier is more costly than that of a simple, linear model and hence requires more memory and runtime. As the training process is an offline process, these factors might not be problematic in real life application scenarios, though.

Figure 4.13 depicts the performance from classification models trained on a subset of 2,000 sentences. Plots on the left side give an overview on the accuracy of joint feature vectors, their counterparts on the right side show the superior quality of joint, binarised feature vectors. Mean ROC area for joint feature vectors ranges from 0.93 for $k = 0$ noisy features down to 0.50 for the maximum noise level of $k = 10{,}000$. Joint, binarised feature vectors show an excellent performance, especially considering the small subset size, and remain near perfect at 0.99 up to a noise level of 1,000. The minimum

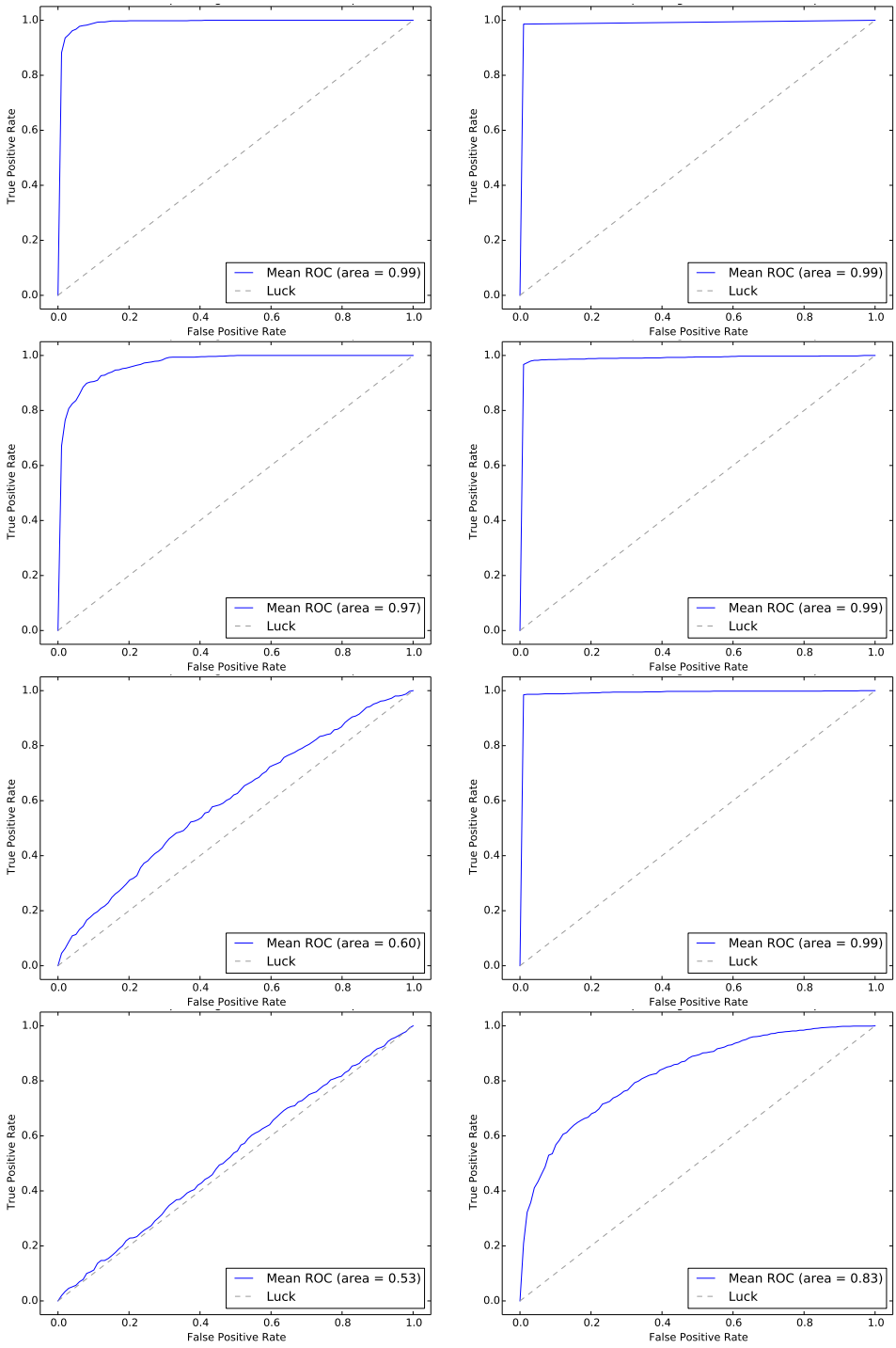


Figure 4.8: **Subset: 2,000, Kernel: linear, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

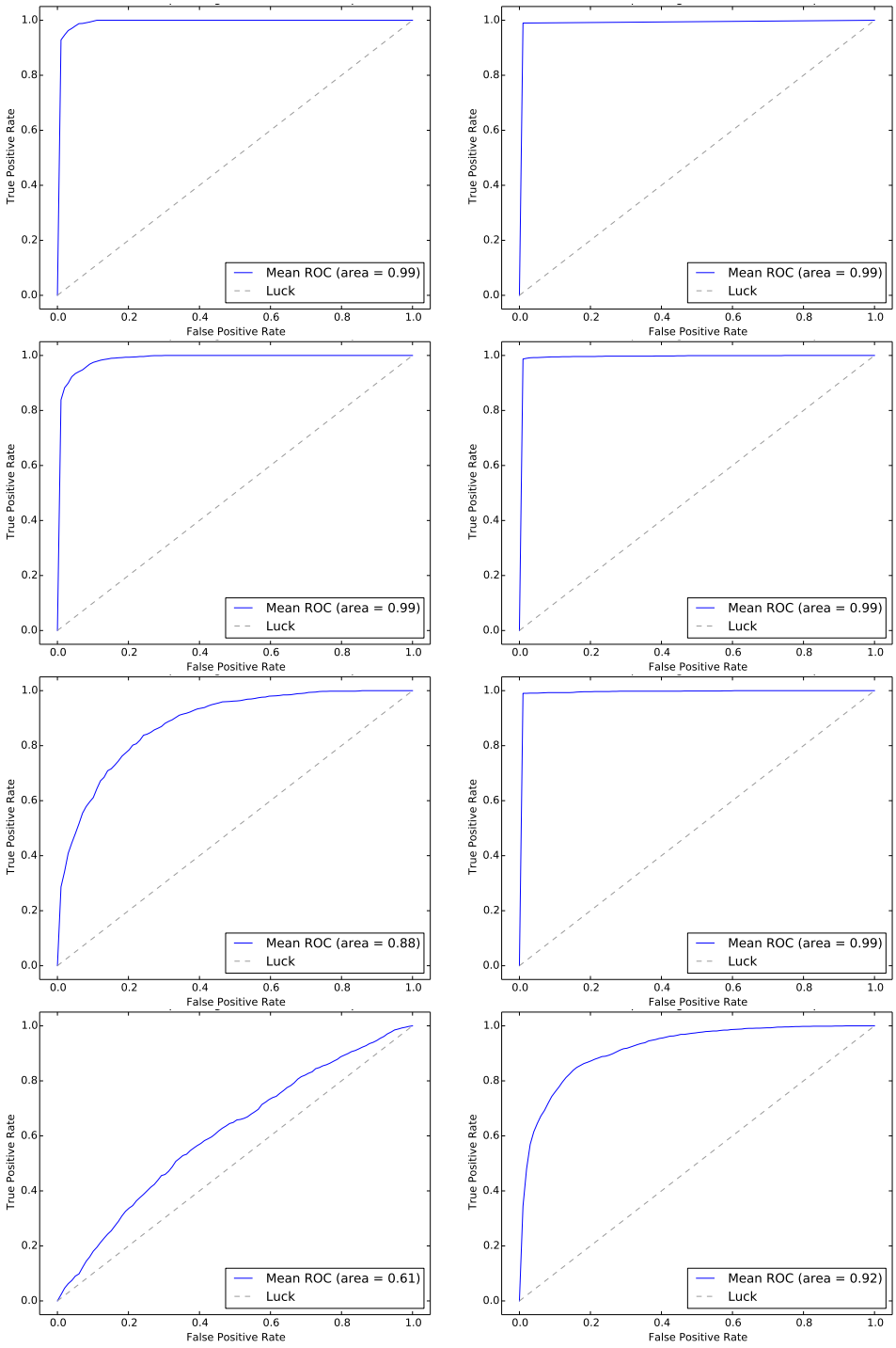


Figure 4.9: **Subset: 4,000, Kernel: linear, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

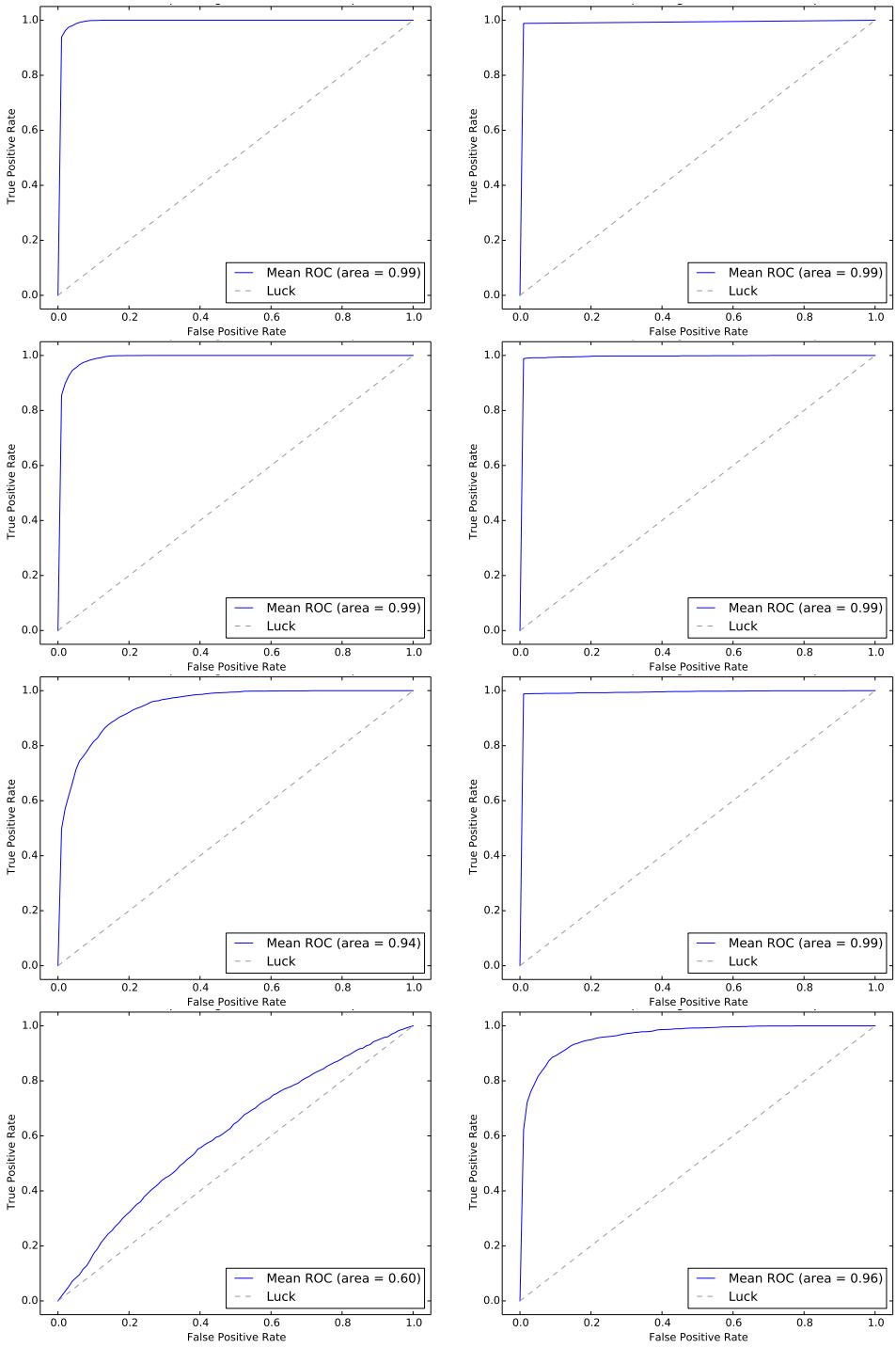


Figure 4.10: **Subset: 6,000, Kernel: linear, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

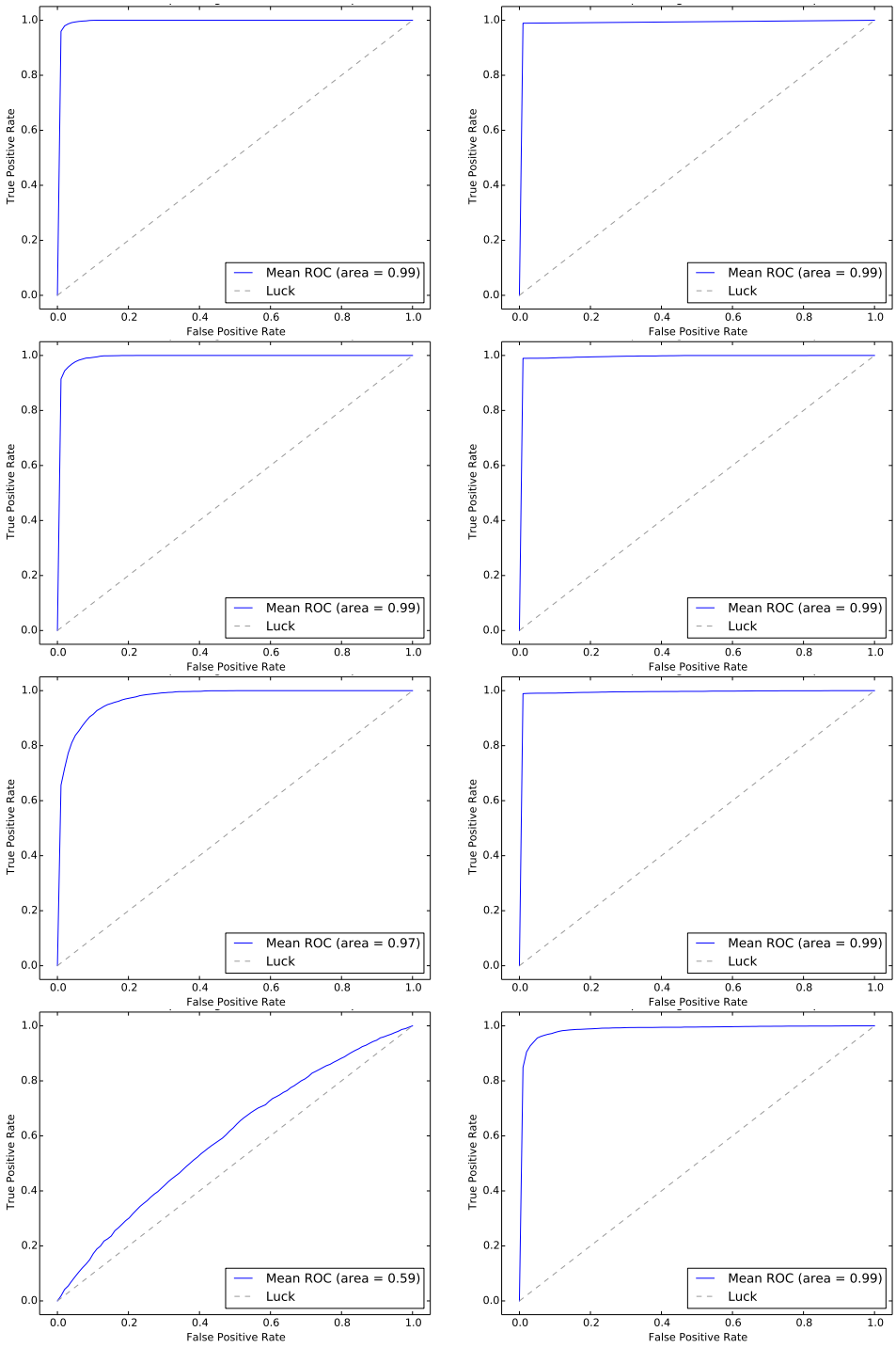


Figure 4.11: **Subset: 9,000, Kernel: linear, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

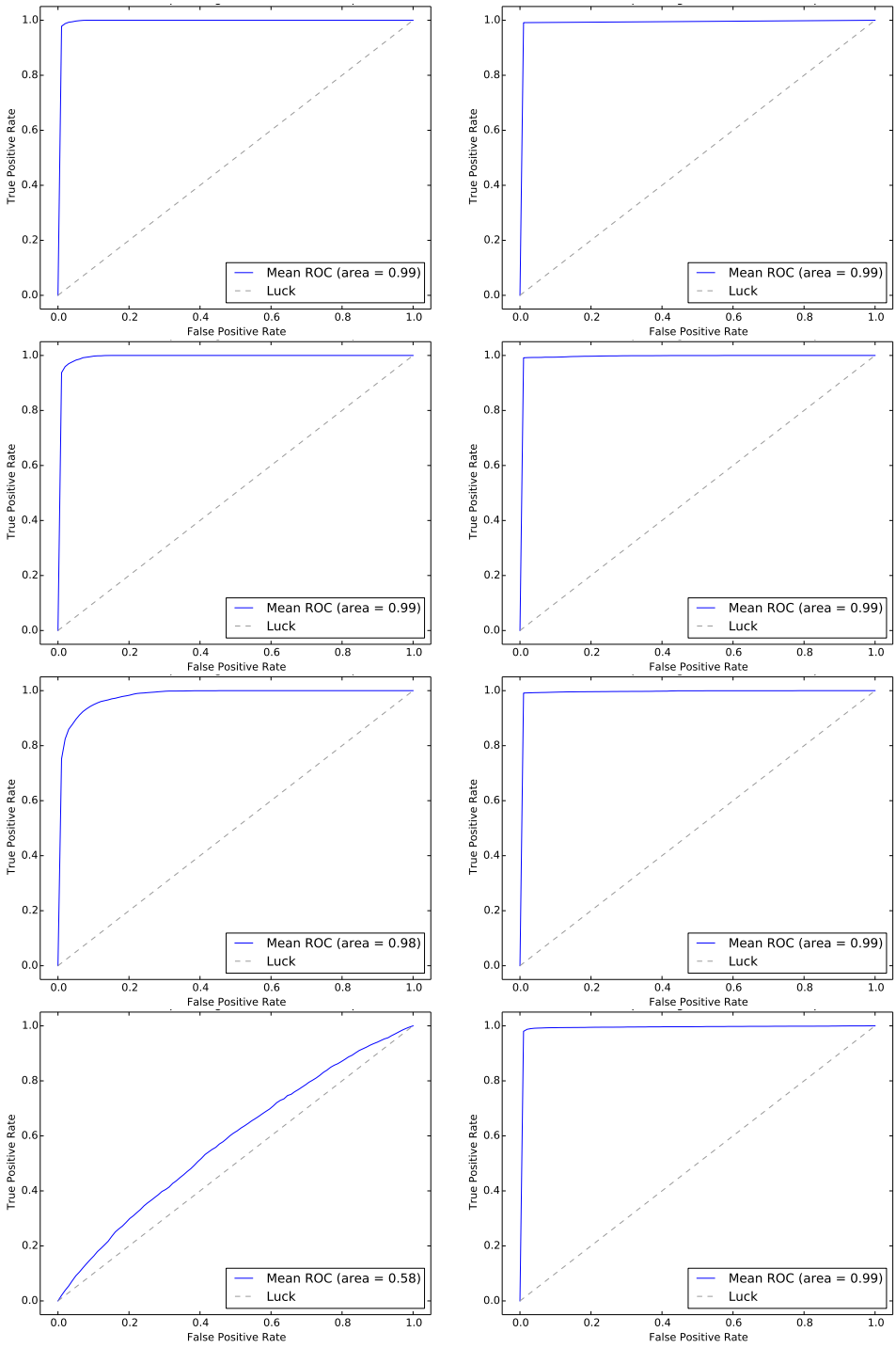


Figure 4.12: **Subset: 12,000, Kernel: linear, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

value is observed at 0.81 for a massive amount of $k = 10,000$ noisy features, which is better than the performance of a classifier trained on joint feature vectors with a much smaller noise level of 100.

This trend continues with increasing subset size. As observed before, prediction accuracy for both feature vector types increases. Joint, binarised vectors perform better, only diminishing to 0.96 for maximum noise level.

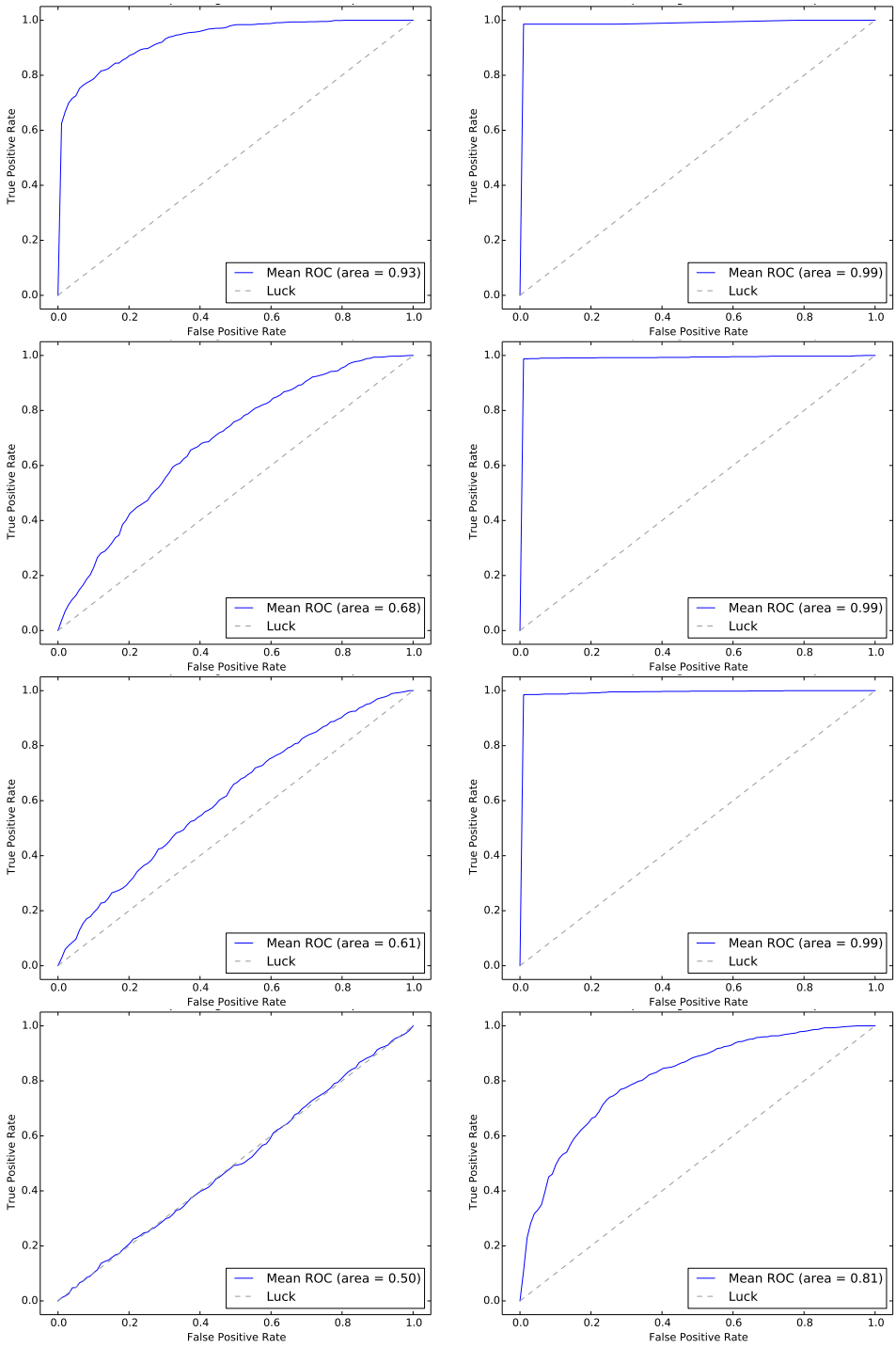


Figure 4.13: **Subset: 2,000, Kernel: rbf, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

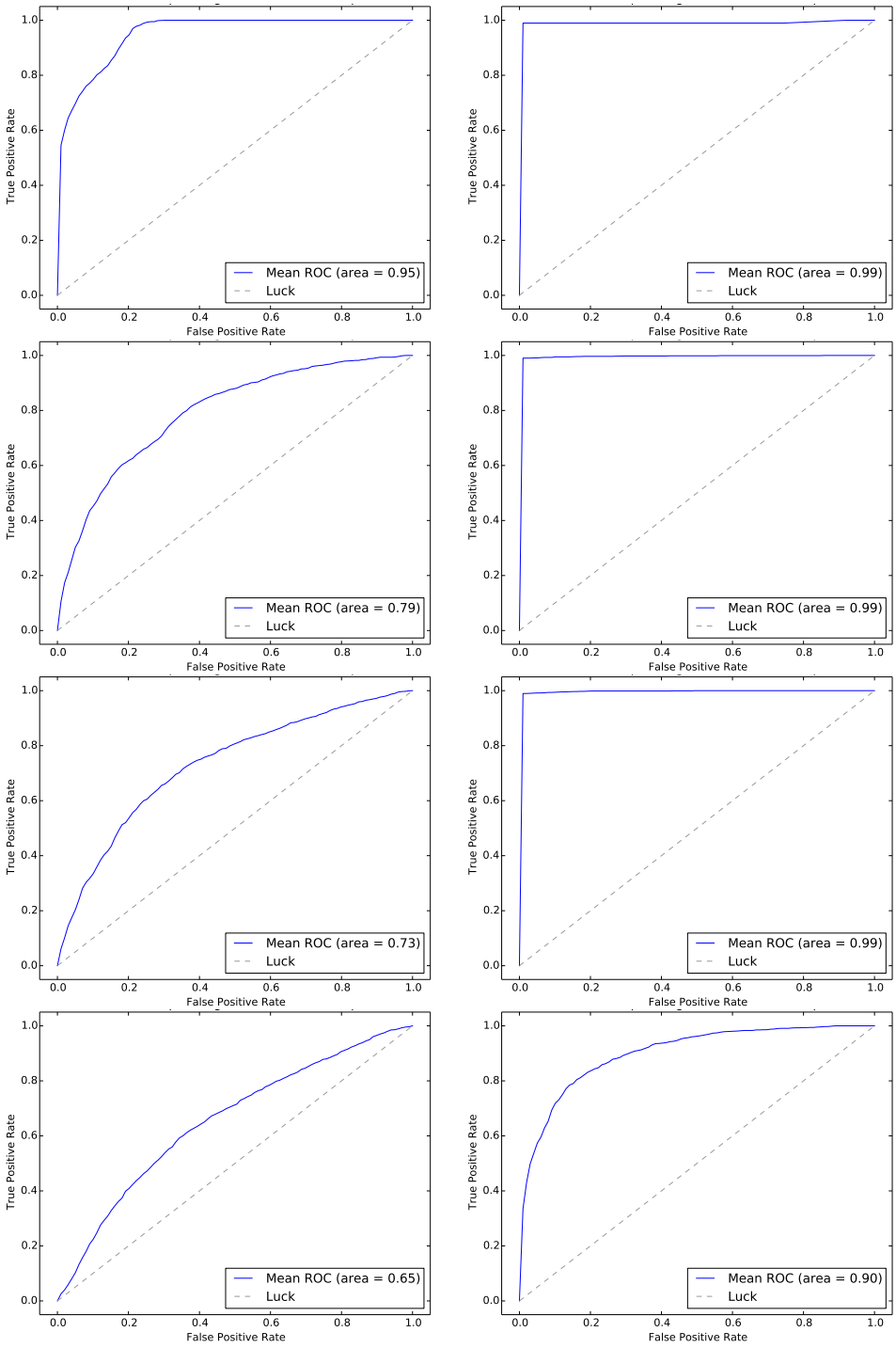


Figure 4.14: **Subset: 4,000, Kernel: rbf, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

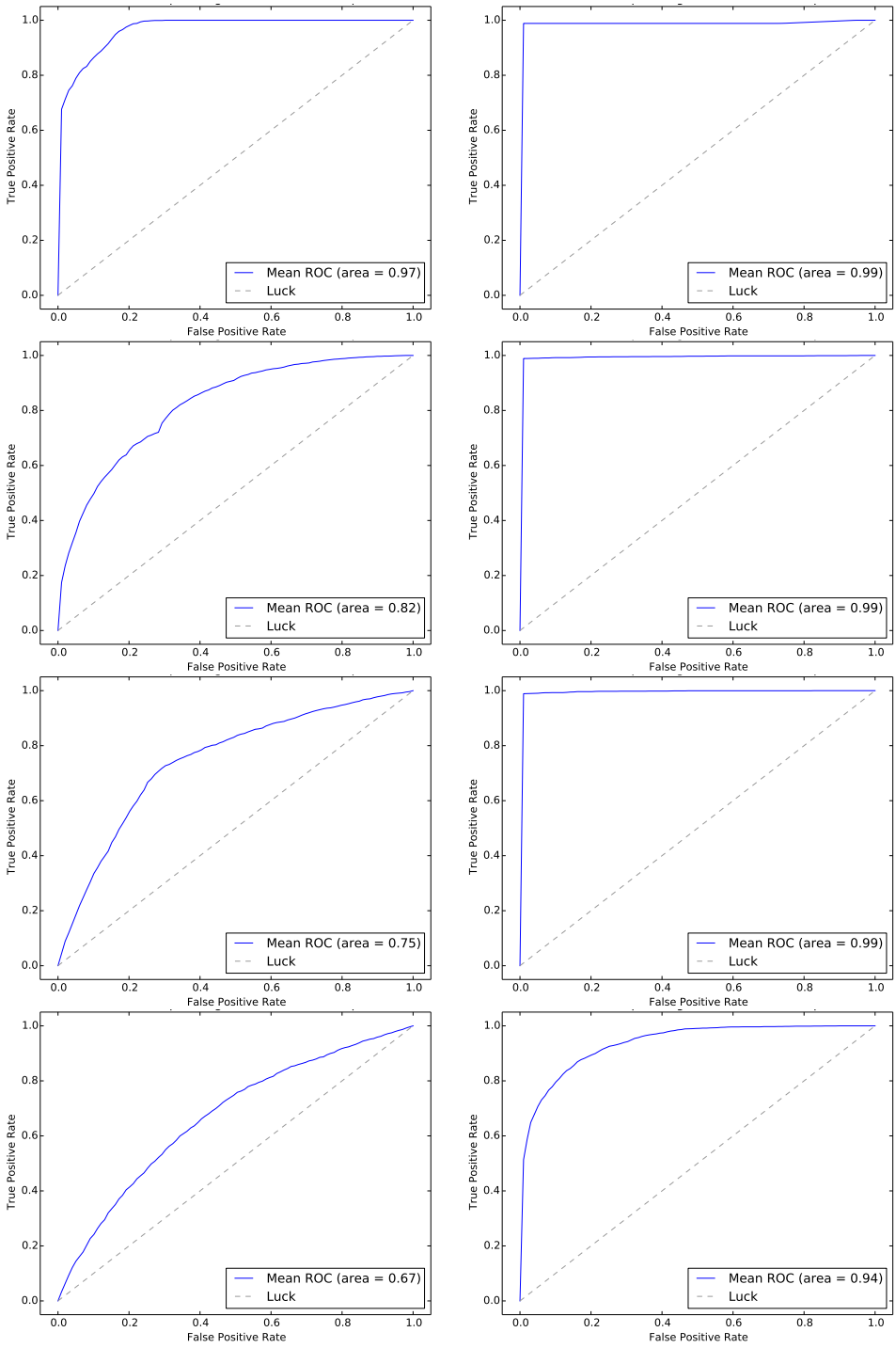


Figure 4.15: **Subset: 6,000, Kernel: rbf, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

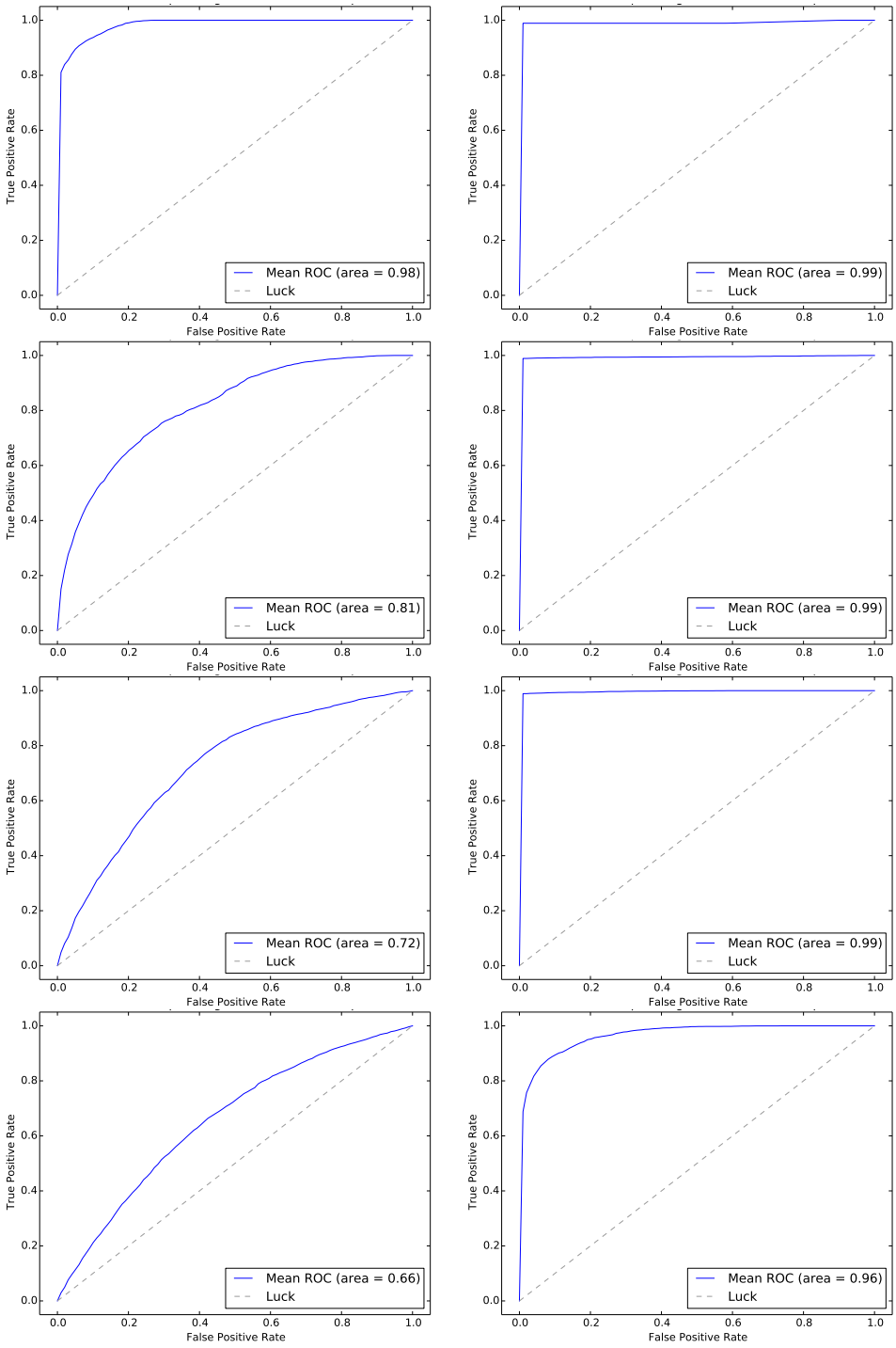


Figure 4.16: **Subset: 9,000, Kernel: rbf, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.

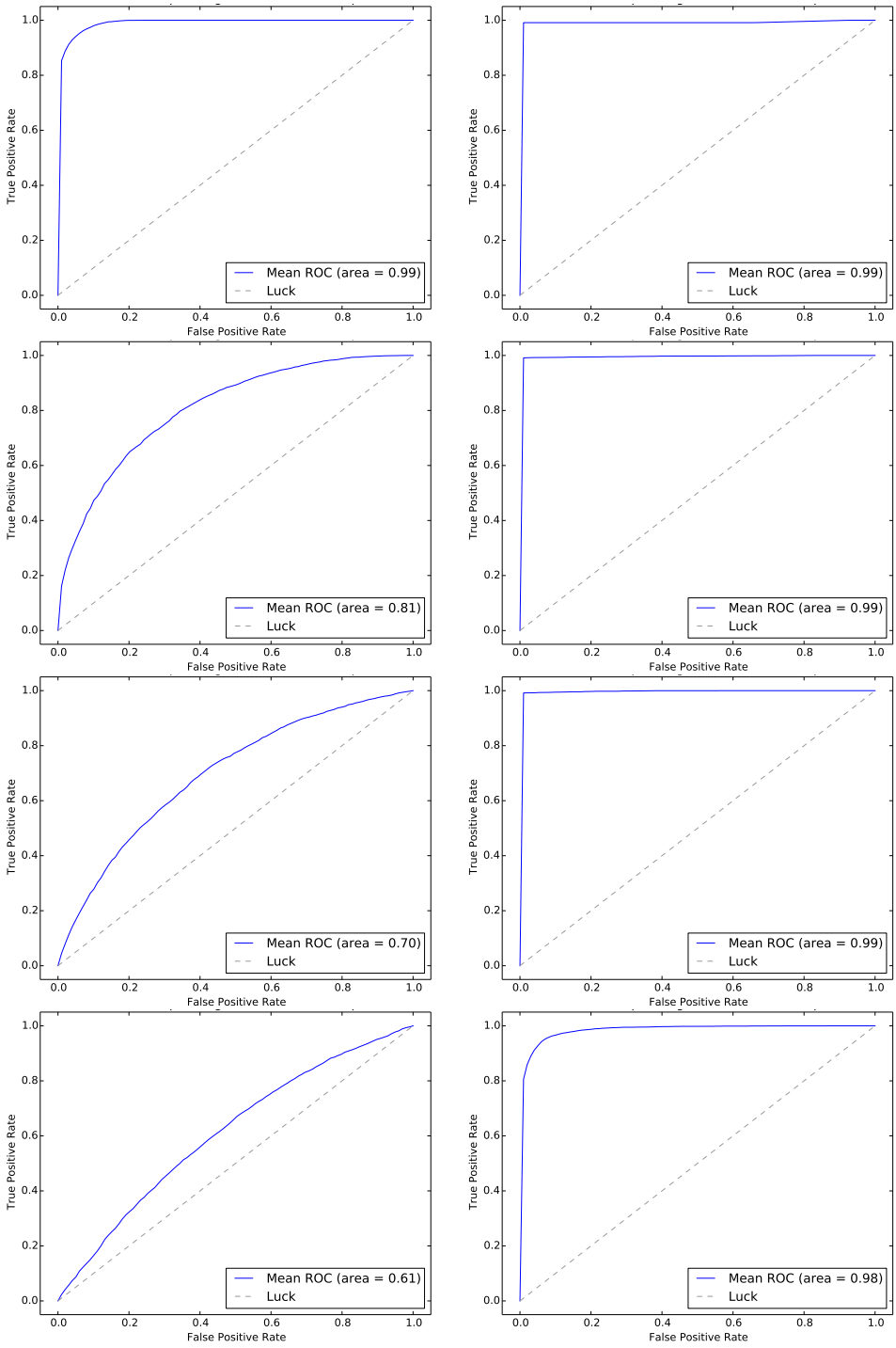
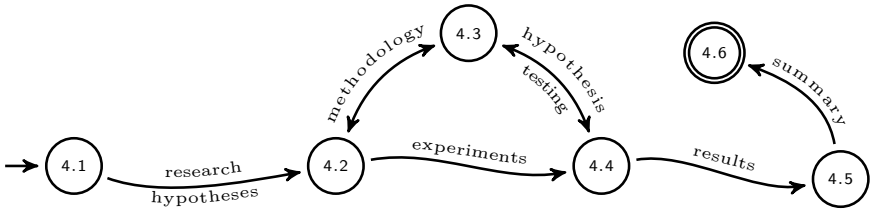


Figure 4.17: **Subset: 12,000, Kernel: rbf, Type: SVC**
Left: accuracy for classification with joint feature vectors with noise levels 0, 100, 1,000, and 10,000 (top to bottom). *Right:* accuracy for corresponding models trained using joint, binarised feature vectors.



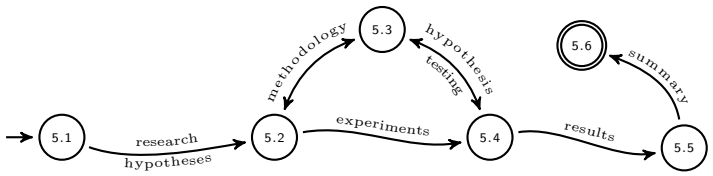
4.6 Chapter Summary

In this chapter we have investigated if joint feature vectors which encode the set of feature values for two candidate systems A, B can outperform single feature vectors only containing features for individual systems. We have stated two research hypotheses and then carefully verified them in a series of experiments, measuring the performance of the binary classifiers in terms of accuracy and time required for model estimation. Similar to the previous chapter, we opted for simulated feature values which have an optimal correlation with the respective target class values.

Our first working hypothesis, *joint feature vectors can outperform single feature vectors*, is confirmed by our experiments. Accuracy of single feature vectors is slightly worse than that of the corresponding models trained on joint feature vectors. Increasing levels of noise diminish accuracy. Still, joint feature vectors are better in terms of training time.

After having verified the usefulness of joint feature vectors, we examined whether the comparison-based binarisation of feature values can improve prediction accuracy of the resulting binary classification models. Results from our experiments confirm this, hence our second hypothesis, *Comparison-based binarisation of feature values can outperform the corresponding joint feature vectors* also holds.

The next chapter presents a framework for system combination using joint, comparison-based binarisation of feature vectors, combining results from Chapters 3 and 4.



5 System Combination Framework

"A good plan violently executed *Now* is better than a perfect plan next week."

– George S. Patton: *War As I Knew It*, 1947.

"What was once thought can never be unthought."

– Friedrich Dürrenmatt: *The Physicists*, 1962.

"No *good* model ever accounted for *all* the facts since some data was bound to be misleading if not plain wrong."

– James D. Watson: as cited by Francis Crick in *Some Mad Pursuit*, 1988.

5.1 Introduction

In previous chapters we have shown that 1) sentence selection approaches have a large potential for system combination of machine translation output (Chapter 3), and that 2) binary classifiers based on joint, comparison-based binarisation of feature vectors present a powerful technique to implement pairwise comparison of candidate systems (Chapter 4). In this chapter, we combine these findings into a framework performing sentence selection to produce hybrid machine translation output. Of course, its selection process

is driven by classification models which are trained using joint, comparison-based binarisation of feature vectors.

Problem Statement

As with previous chapters, we first formulate an overall problem statement to properly define the matter of investigation. As the fundamental parts of our system combination approach have already been defined and discussed, we can now focus on bringing the individual pieces together. The ultimate goal is to propose a machine learning framework for hybrid machine translation. This gives the following problem statement:

> *Given that sentence selection seems to have a promising potential and further considering the positive performance of joint, comparison-based binarisation of feature vectors, can we combine these into a competitive system combination framework for machine translation output?*

Following the line of argumentation first introduced in Chapter 1, we want to implement a *sentence-selection-based approach* in order to preserve both syntactic and semantic properties of the chosen translation output. The obvious drawback in comparison to, e.g., confusion networks is that such a system can never generate combination output which is "fused" from good sub-segments of several candidate translations. Our methodological paradigm can never outperform the translation quality of the single-best candidate translation which is contained in the set of input sentences.

On the other hand, in Chapter 3 we have shown that, even without the potential of combining phrase-level properties of the individual candidate translations, oracle-based sentence selection has an impressive potential. In-

terestingly, even globally bad systems may contribute a fraction of good translations. Furthermore, it is conceivable that, in future work, our combination framework can be extended to also support combinations on the phrase level, e.g., by choosing one system as the translation "template" which may choose phrase translations from other candidate systems using binary classifiers.

This chapter is structured as follows: in Section 5.2 we state our research hypothesis. Second, we describe the methodology (Section 5.3) we have used in our experiments which are discussed afterwards (Section 5.4). Finally, we present results (Section 5.5) and some examples before ending with a summary of our findings and a conclusion in Section 5.6.

5.2 Research Hypothesis

First let us define the research hypothesis of this chapter. Based on the assumption that sentence selection on the document level can be used to combine multiple candidate translations into a new, combined translation, we want to investigate whether an implementation of such an approach which is based on binary classification models trained using joint, comparison-based binarisation of feature vectors can be defined and how successful it can be. In essence, we are trying to verify that the fundamental findings of Chapters 3 and 4 can be turned into a competitive combination approach.

Of course, the creation of such a combination system is only the first step. Once trained, we have to compare its performance against other system combination approaches. Specifically, we are interested in the comparison to state-of-the-art methods based on confusion networks. If you recall our motivation in Chapter 1, such approaches account for the majority of system combination research in the last decade. Hence, they are a baseline we have to compare our method against in order to rightfully claim that a sen-

tence selection approach trained with joint, comparison-based binarisation of feature vectors is competitive.

This leads to the following research hypothesis:

Hypothesis 6. *It is possible to define and implement a sentence selection approach for system combination which is based on binary classification models trained with joint, comparison-based binarisation of feature vectors that is competitive with state-of-the-art system combination methods.*

We verify this hypothesis in the remainder of this chapter.

5.3 Methodology

Motivation

Our methodology utilises machine learning to estimate binary classification models which can be used to solve a sentence selection problem, namely to identify the *optimal translation* for a given source sentence. In order to reduce the complexity of this problem, we use a *divide and conquer* approach and define an algorithm which computes the best translation by considering all pairwise comparisons of systems A, B. This further has the advantage that binary classification is a very well studied area of machine learning, meaning that there exist a lot of techniques to optimise the classification models used for our selection task. In fact, we are flexible regarding the actual machine learning paradigm which is used for training the classifiers. Based on the results from Chapter 4 we apply comparison-based binarisation on the feature values for both systems A, B, effectively making use of joint, comparison-based binarisation of feature vectors during model estimation. Following the train of thought of our thesis research, this will allow us to efficiently handle the underlying classification problem.

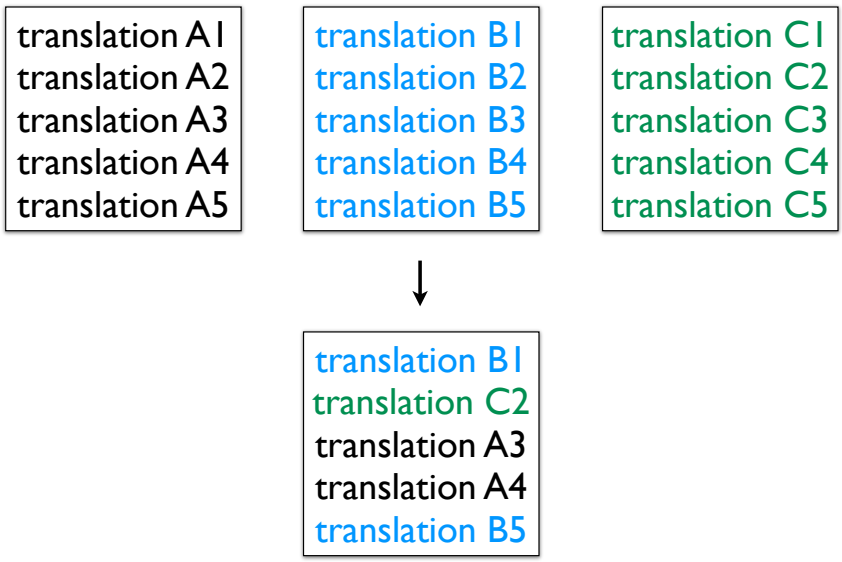


Figure 5.1: Schematic overview of our sentence-selection-based combination approach. Given a set of k translations for some N translation engines, we compute the final translation by selecting the "best" candidate translations. Selection is based on the results of a machine learning classifier.

Figure 5.1 illustrates the problem we are faced with. Given translation output (at the document level) from several machine translation engines, we want to create a combined translation by choosing the best translation option per sentence. Our method applies machine learning to train a classifier that then can be used to perform candidate selection on the sentence level.

We first want to provide an informal overview of our methodology for hybrid machine translation. We use the terms *hybrid machine translation* and *system combination* interchangeably in this thesis. Our architecture is based on classifiers trained using state-of-the-art machine learning tools. We require the availability of a set of n translations from several MT systems that are treated as "black boxes", meaning that we can only access the individual systems' translation output but do not have means to extract *system-internal features* such as scores, preferences, or similar properties of the translations.

For training of the binary classification model (*or models as we can train individual classifiers for all pairwise comparisons of systems A, B*) we need training and development sets including the corresponding reference text. Without references we cannot order candidate translations according to their respective quality to label the training data which is a mandatory step in supervised machine learning.

Using the development data (consisting of candidate translations from n translation systems and the corresponding source and reference texts) we perform the following processing steps to generate a hybrid translation for some given test set:

1. Compute a total order of individual system output on the development set using some order relation based on quality assessment of the translations with automatic metrics. This can also be extended to include results from manual evaluation (though it may be costly to actually add human judgment to the overall process);
2. Decompose the aforementioned system ranking into a set of pairwise comparisons for any two pairs of systems A, B. As we do not allow for ties in our system comparisons[1], the two possible values $A > B$, $A < B$ also represent our *machine learning classes* $+1/-1$, respectively;
3. Annotate the translations with feature values derived from NLP tools such as *language models*, *word alignment models*, *lexical phrase tables*, *part-of-speech taggers*, or *parsers*;
4. Create a data set for training a machine learning classifier which is able to estimate (based on the features) which of two given systems A, B is

[1] Of course, the possibility that translations for two systems A, B are equal according to the chosen quality metrics exists. In such cases we would either drop the corresponding system pair from our set of training instances or define one of the systems to be the winner, e.g., by assigning target class $+1$ for all values $A \geq B$. One can also use *ternary* target classes $+1/0/-1$.

better according to the available features;

5. Train such a machine learning classification model using, e.g., 1ibSVM, see [Chang and Lin, 2011];

Steps 1–5 represent the *training phase* in our framework. The availability of a training set including references is required as this is needed to allow the definition of an ordering relation which subsequently defines the training instances for the machine learning framework of choice. After training, we can use the resulting binary classifier as follows:

6. Apply the resulting classification model to the candidate translations from the given test set. This will *predict* pairwise estimates $+1/-1$ for each possible pair of systems A, B;
7. Perform *round-robin system elimination* to determine the best system from the set of candidate translations for each of the segments. It may be necessary to resolve conflicts at this stage, e.g., if two or more systems achieved the same number of +1 "wins" during the previous step;
8. Using this data, synthesise the final, hybrid translation output.

Steps 6–8 represent the *decoding phase* in which the trained classification model is applied to a set of *unseen* translations without any reference text available. By computing pairwise *winners* for each possible pair of systems and each individual sentence of the test set, we determine the single best system on the sentence level.

As this allows to integrate good translations from otherwise bad systems, we expect the methodology to improve over its individual source systems; provided the choice of features used in such experiments has a good enough correlation with the target class labels. Notably, there are two prerequisites that are necessary in order to make the proposed approach work:

– We need to define a sufficiently good order relation; and
– Based on this order we have to train a binary classification model.

We address both issues later in this chapter.

Fundamentals

Before we can dive into the details of the hybrid combination framework, we first have to define several fundamental concepts, notations, and operators.

Our method is based on machine learning techniques which solve binary classification problems. The classifiers are based on models learnt on the annotation output obtained from *feature functions*. These are defined as follows:

Definition 7 (Feature function)**.** *A **feature function** f takes some input $i \in I$ and computes so-called **feature values** or **feature scores** $x \in X$, mapping from input domain I into output range X. Formally:*

$$f : I \rightarrow X$$

Notation 3 ($feature_{f,X}$)**.** *For convenience, we define the following notation to denote that we compute a feature value x in output range X from some input i:*

$$feature_{f,X}(i) = f(i)$$

If the actual feature function f does not matter in the given context, we may choose to omit it and just use $feature_X(i)$ instead. This denotes that "some" feature function f' is used to compute feature values from output range X.

Notation 4 (Feature value)**.** *The output values of a feature function are called **feature values** or **feature scores**. We use both terms interchangeably throughout the thesis document.*

Notation 5 (Feature space)**.** *The output range of some feature function is referred to as the corresponding* ***feature space****. Any such feature space can either be* ***partially*** *or* ***totally ordered****.*

A feature space is formally defined like this:

Definition 8 (Partially ordered feature space)**.** *A* ***feature space*** *X is a set with an* ***ordering relation*** *$\leq_X$ that is:*

- ***reflexive****: i.e., $\forall a \in X : a \leq_X a$;*
- ***transitive****: i.e., $\forall a, b, c \in X$: if $a \leq_X b$ and $b \leq_X c$ then $a \leq_X c$;*
- ***antisymmetric****: i.e., $\forall a, b \in X$: if $a \leq_X b$ and $b \leq_X a$ then $a = b$.*

Such an ordering relation defines a ***partially ordered feature space****. Note that, as the ordering relation is only partial, there can be elements $a, b \in X$ which cannot be compared using $\leq_X$. Such elements are called* ***incomparable****.*

Definition 9 (Totally ordered feature space)**.** *If the ordering relation $\leq_X$ of a feature space as defined in Definition 8 is also:*

- ***total****: i.e., $\forall a, b \in X : a \leq_X b$ or $b \leq_X a$.*

it defines a ***totally ordered feature space*** *instead. The difference is that the total order guarantees that all elements $a, b \in X$ can be compared using $\leq_X$ and therefore are* ***comparable****.*

Definition 10 (Selection operator Δ)**.** *The* ***selection operator*** *Δ for feature values takes two feature scores and computes which of these is "better" in the respective feature space.*

We define the operator as follows:

$$\Delta : X \times X \rightarrow \{\perp\} \cup \{-1, 0, 1\}$$

The output of the Δ operator is defined like this:

$$\Delta(feature_X(A), feature_X(B)) = \begin{cases} 1 & feature_X(A) > feature_X(B) \\ 0 & feature_X(A) = feature_X(B) \\ -1 & feature_X(A) < feature_X(B) \\ \perp & \text{features scores are incomparable} \end{cases}$$

where the given feature values $feature_X(A), feature_X(B)$ are both elements of the same feature space X. The comparison result of the two scores is encoded using integers taken from $\{-1, 0, 1\}$ if they are comparable under $\leq_X$. Otherwise we use $\perp$ to denote that the feature values are incomparable. Note that our definition of totally ordered feature spaces guarantees that all feature scores $a, b \in X$ are comparable under $\leq_X$, effectively changing the operator's signature to:

$$\Delta : X \times X \rightarrow \{-1, 0, 1\}$$

For natural or real numbers, this comparison typically means evaluating the sign of the difference between the given feature values, $d = feature_X(A) - feature_X(B)$, effectively computing $\Delta(feature_X(A), feature_X(B))$ as $sgn(d)$.

Definition 11. *The* classification operator Γ *takes the comparison results of N feature values for two individual candidate translations and computes which of these is better according to the underlying model M.*

Formally, the operator is defined as follows:

$$\Gamma_M : \{-1, 0, 1\}^N \rightarrow \{+1, -1\}$$

The output of the Γ operator is based on the model's prediction, formally:

$$\Gamma_M(f_1, f_2, \ldots, f_N) = M.predict(f_1, f_2, \ldots, f_N)$$

Ranking Candidate Translations

Notation 6 (Corpus-level Metric)**.** *A quality assessment metric M which has been optimised for correlation on the corpus level is called a* ***corpus-level metric****. We use subscript $_C$ to denote this, e.g., $Meteor_C$ refers to the Meteor metric being used on corpus level.*

Notation 7 (Sentence-level Metric)**.** *Similarly, quality metrics can also have been optimised for correlation on the sentence level. We use subscript $_S$ to denote such a* ***sentence-level metric****, e.g., $Meteor_S$ refers to the Meteor metric being used on the sentence level.*

In order to rank the given candidate translations, we first have to define an *ordering relation* over the space of translation outputs. For this, we apply the following evaluation metrics which are the *de-facto standards* for automated assessment of machine translation quality. We consider:

1. The Meteor score as described in [Denkowski and Lavie, 2011]. We work with scores from the sentence and from the corpus level;
2. The NIST n-gram co-occurence score, published by [Doddington, 2002], on the corpus level; and
3. The BLEU score as defined in [Papineni et al., 2002] which is the most widely used evaluation metric in MT, used on the corpus level.

While both BLEU and NIST scores are designed to have a high correlation with results from manual evaluation on the corpus level, the Meteor metric can also be used to compare translations on the level of individual sentences.[2] We use this property when defining our order $ord(A, B)$ on translations, as shown in equations 5.1 and 5.2.

$$ord(A, B) \stackrel{\text{def}}{=} ord_X(A, B) \tag{5.1}$$

[2]There exist other metrics which produce usable output on the sentence level, e.g., s-BLEU. These could also be used instead of Meteor.

where $X \in \{Meteor_S, Meteor_C, NIST_C, BLEU_C, \perp\}$. Suffix $_S$ denotes a *sentence-level* quality metric score while suffix $_C$ represents a *corpus-level* score. $\perp$ denotes the "empty" metric which is the "minimal" element in the set of available metrics.

$$ord_X(A,B) \stackrel{\text{def}}{=} \begin{cases} +1 & \text{if} X_A > X_B \\ -1 & \text{if} X_A < X_B \\ ord_{X'}(A,B) & \text{otherwise}, X > X' \end{cases} \tag{5.2}$$

If candidate translations A, B are indistinguishable by current metric X, we *recursively* delegate the decision problem to metric X' where $X' < X$ denotes the "next-best" metric in our (ordered) set of available metrics: $Meteor_S$, $Meteor_C$, $NIST_C$, $BLEU_C$. By definition,

$$ord_\perp(A,B) = 0 \tag{5.3}$$

which means that candidate translations A, B are of equal translation quality according to the quality metrics used. Pairs of systems with $ord(A,B) = 0$ can either be removed from the set of instances used for training the machine learning classifier or we can fall back to using a *pre-defined fallback strategy* that decides which of the two systems is supposed to be better.

Algorithm 3 on page 133 presents an algorithm for computing a ranking between N candidate systems, based on the output of an ordering relation ord_X. We initialise in line 1. Afterwards, lines 2–4 compute automated metrics' scores which are used by ord_X to perform the pairwise comparison of candidate translations. *Ranking items* (a, b, i, o) encode which of the two systems a, b is better, according to ord_X, for segment i. System a is better for $o = +1$, system b if $o = -1$. Otherwise, systems are equal according to ord_X. They are computed from line 5 to 10 for all pairs of systems a, b and segments i. In line 11, we return the set of ranking items.

Feature Binarisation

Given a set of feature vectors from N candidate systems, we need to compute joint, binarised feature vectors for all pairwise comparisons of systems A, B. Algorithm 4 on page 134 explains how we can compute such joint, binarised feature vectors based on single feature vectors. First, we initialise in line 1. Then, we iterate over all pairs of feature vectors F_a, F_b. For each of these pairs, we loop over the set of individual segments and, finally, compare each of the feature values using our selection operator Δ. The resulting value $\delta \in \{\perp\} \cup \{-1, 0, 1\}$ represents the binarised "variant" of the two corresponding feature values $F_{a,i,f}$ and $F_{b,i,f}$. We return a set of *binarised feature items* (a, b, i, f, δ) in line 10. These encode which of the two systems a, b is better in terms of feature f extracted from segment i. A δ value of +1 denotes that system a is better in this respect. The same holds for system b if $\delta = -1$. Equal feature values result in a value of 0, incomparable features force a $\perp$ result. Depending on whether we aim for a binary or a ternary classification model, equal feature values may have to be removed. Incomparable feature values should always be removed before we can start training a classification model on our binarised features.

Classifier Estimation

We can now compute 1) sets of ranking items and 2) sets of binarised feature items. The former encodes the ordering of all pairs of candidate systems A, B for all segments i of the training set, the latter allows to generate joint, binarised feature vectors from which a binary classification model can be trained. In machine learning terms, the set of ranking items is equivalent to the *target class values* y while the set of binarised feature values roughly corresponds

to the *training vectors* X. Given a machine learning method **train-classifier**[3] and the two data sets, the estimation of a binary classifier is straightforward. Algorithm 5 describes the estimation process in detail.

We initialise our data structures in lines 1–2. X collects training vectors in joint, binarised format and y stores target class values $y \in \{-1, +1\}$. We iterate over all pairs of systems a, b and all segments i in lines 3–4. In lines 5–8, we compute the corresponding joint, binarised feature vector. We update our training vector and target class sets in lines 9 and 10. The classification model is then estimated with **train-classifier** in line 13 and returned in line 14.

System Combination using Binary Classification

Assuming we have successfully trained a binary classification model M based on joint, comparison-based binarisation of feature vectors as introduced in the Chapter 4, we can generate a combined translation from N translation engines by selecting the best candidate translation per sentence, guided by the decisions made by our classifier M. While this sounds simple, we have to consider the prediction rate of the underlying classification model—it has a direct influence on the translation quality of the resulting output. The closer our model's prediction accuracy is to random guessing, the less likely it is that our system combination approach functions properly. Therefore it is recommended to optimise classification models during training.

Our sentence selection mechanism is faced with the following problem: given candidate translations from N translation engines, it has to apply the ΓM operator to all possible, pairwise comparisons of candidate translations and then determine the "best" translation available for each of the input sentences. Algorithm 6 on page 136 shows how system combination based on

[3] This can be, e.g., the `fit` method of a `svm.SVC` or a `svm.LinearSVC` object instance, when implementing estimation with scikit-learn.

binary classification is implemented. A schematic overview of the selection problem on segment level is depicted in Figure 5.2.

Running $\Gamma_M(A, B)$ for some given binary classification model M and all pairwise comparisons of N candidate translations, memorising the number of wins per system, can result in two different outcomes:

1. There is a unique winner. This is depicted in Figure 5.3; or
2. Several systems have an equal number of wins. This means that we have to apply a conflict resolution strategy in order to identify the final winning system. See Figure 5.4.

We explain both scenarios in the following section.

Case 1: Single Winner

If a single candidate translation is identified as the unique winner for the given input sentence, we simply copy it over to the combined translation, as shown in Figure 5.3, and then proceed to the next sentence.

Case 2: Multiple Winners

If we have observed the best number of wins for two or more systems, we have to apply a conflict resolution strategy to identify the best system for the current input sentence. The following sub-cases need to be considered:

- There are two best systems X, Y;
- Three or more systems are in the set of best systems.

The first sub-case is easy to handle. Due to the nature of the implementation of our machine learning classifier, we are guaranteed to receive a clear-cut decision for each possible, pairwise comparison of systems as our classifier will

return either +1 or −1. Also, by the design of our sentence selection mechanism, we know that we have already computed all these pairwise comparison decisions. Hence, we can simply look up the results of $\Gamma_M(X, Y)$ and choose the winner of this comparison as the overall best system for the current input sentence.

The second sub-case is a more problematic. While it may be possible that one of the systems has been judged as better than all the other systems available in the set of winning translation engines for the current input sentence, this is a property which is in no way guaranteed as there might be *circular dependencies* such as, e.g., $X > Y$, $Y > Z$, and $Z > X$, which cannot be resolved so easily.

If one such system X can be identified, it becomes the winner of this comparison as the overall best system for the current input sentence and we are done. If this distinction is not possible, as is the case for circular dependencies, we have to fall back to *pre-defined* knowledge from the training phase and choose one of the candidates as "the best option".

Algorithm 3 Computing a ranking for N candidate systems

Require: set of translations from N candidate systems $S = \{S_1, S_2, \ldots, S_N\}$;
Require: reference text R;
Require: ordering relation ord_X, considering metrics defined in X.
Ensure: $|S_1| = |S_2| = \ldots = |S_N| = |R|$

$metrics \leftarrow \emptyset, ranking \leftarrow \emptyset$

for each system $S_a; 1 \leq a \leq N$ **do**
    $metrics \leftarrow$ **compute-metric-scores**(S_a) ▷ Metric scores are required for ord_X
end for

for each pair of systems $S_a, S_b; 1 \leq a < b \leq N$ **do**
    for each segment id $i; 1 \leq i \leq$ # of segments **do**
        $o \leftarrow ord_{\text{metrics}}(S_{a,i}, S_{b,i})$ ▷ Rank systems for current segment, $o \in \{-1, 0, 1\}$
        $ranking = ranking \cup (a, b, i, o)$ ▷ Store resulting ranking item
    end for
end for

return $ranking$ ▷ Return full set of ranking items

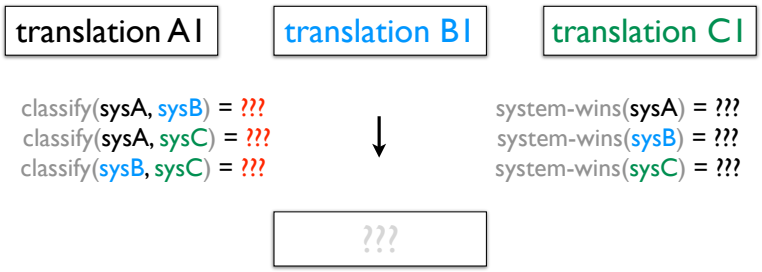


Figure 5.2: Problem for the sentence selection mechanism: given a set of N translation candidates, select the "best" translation, according to the binary classification model, which is available for the current input sentence.

Algorithm 4 Feature binarisation for N candidate systems

Require: set of feature vectors from N candidate systems $F = \{F_1, F_2, \ldots, F_N\}$;
Require: selection operator $\Delta : X \times X \rightarrow \{\perp\} \cup \{-1, 0, 1\}$.
Ensure: $|F_1| = |F_2| = \ldots = |F_N|$

$binarised \leftarrow \emptyset$

for each pair of feature vectors $F_a, F_b; 1 \leq a < b \leq N$ **do**
  for each segment id $i; 1 \leq i \leq$ # of segments **do**
    for each feature id $f; 1 \leq f \leq$ # of features **do**
      $\delta \leftarrow \Delta(F_{a,i,f}, F_{b,i,f})$ ▷ Compare current feature values, $\delta \in \{\perp\} \cup \{-1, 0, 1\}$
      $binarised = binarised \cup (a, b, i, f, \delta)$ ▷ Store resulting binarised feature item
    end for
  end for
end for

return $binarised$ ▷ Return full set of binarised items

5.4 Experiments

Features for Classifier Estimation

We have experimented with many different feature values during the research for this thesis. Irrespective of the actual approach that is implemented in a given machine translation system, the creation of its translation output usually requires several, often heterogeneous, features. These can be:

- simple scores, e.g., for language model, parser, or lexical phrase table probabilities;
- more complex data structures such as hierarchical parse trees or word alignment links; or
- even full parse forests or n-best lists containing both translations and corresponding feature values.

Algorithm 5 Classifier estimation for N candidate systems

Require: set of binarised feature items $binarised$;
Require: set of ranking items $ranking$;
Require: machine learning training method **train-classifier**.
Ensure: $|binarised_{a,b,i}| = |ranking_{a,b,i}|; 1 \leq a < b \leq N; 1 \leq i \leq$ # of segments

$X \leftarrow \emptyset$ ▷ Stores binarised feature vectors for pairwise comparisons
$y \leftarrow \emptyset$ ▷ Store corresponding target class values +1/−1

for each pair of system ids $a, b; 1 \leq a < b \leq N$ **do**
    for each segment id $i; 1 \leq i \leq$ # of segments **do**

        $features_{a,b,i} \leftarrow \emptyset$

        for each feature id $f; 1 \leq f \leq$ # of features **do**
            $features_{a,b,i} = features_{a,b,i} \cup binarised_{a,b,i,f}$
        end for

        $X = X \cup features_{a,b,i}$
        $y = y \cup ranking_{a,b,i}$
    end for
end for

$classifier \leftarrow$ **train-classifier**(X, y) ▷ Estimate classifier on joint, binarised feature vectors

return $classifier$ ▷ Return binary classification model

Given this wide range of features and their diversity, it is very difficult to get an *intuitive* understanding of the inner workings of the MT engine in question; thus, further research work on the combination of machine translation systems into better combination systems seems to be of high importance to the field. To overcome the aforementioned problems with incomprehensible feature values, we have proposed a method that is driven by machine learning tools, hence leaving both the exact interpretation and weighting of features to the machine learning algorithms used, relying on their discrimi-

Algorithm 6 System combination using a binary classification model M

Require: binary classification model M;
Require: set of binarised items $binarised$ for some test set;
Require: classification operator Δ_M.
Require: machine learning training method **train-classifier**.

$result \leftarrow \emptyset$ ▷ Stores best system per segment

for each segment id $i; 1 \leq i \leq$ # of segments **do**
  $wins \leftarrow \emptyset$ ▷ Stores number of wins per system

  for each pair of system ids $a, b; 1 \leq a < b \leq N$ **do**
    $features_{a,b,i} \leftarrow \emptyset$
    for each feature id $f; 1 \leq f \leq$ # of features **do**
      $features_{a,b,i} = features_{a,b,i} \cup binarised_{a,b,i,f}$
    end for

    $prediction = \Delta_M(features_{a,b,i})$ ▷ Predict target class
    if $prediction == +1$ **then**
      $wins \leftarrow wins \cup a$ ▷ System a wins
    else
      $wins \leftarrow wins \cup b$ ▷ System b wins
    end if
  end for

  $result = result \cup$ **compute-winner**$(wins)$
end for

return $result$ ▷ Return set of best systems per segment

native power which eventually will find weights that can help to create combined translations from the given candidate translations with a good quality. We did not deeply investigate feature engineering and optimisation for the results reported in this Chapter, leaving that to future work.

We create the training data set for classifier estimation using a selection of *features*. Feature values are extracted from the given candidate translations

using external tools which are applied to all candidate systems. This allows to extract feature values which are comparable between candidates. While there are many possible features which could be added to our feature set, we mostly focused on the following choice, leaving more focused research on feature values for system combination to future work:

- number of target tokens;
- ratio of target/source tokens;
- number of target parse tree nodes;
- ratio of target/source parse tree nodes;
- number of target parse tree depth;
- ratio of target/source parse tree depth;
- n-gram language model score for order $n \in \{1,\ldots,5\}$;
- language model perplexity for order $n \in \{1,\ldots,5\}$.
- lexical translation probabilities from phrase tables.[4]

These features represent a combination of (shallow) parsing, language model scoring as well as machine translation techniques and are derived from the set of features that are most often used in the respective system combination literature such as, e.g., [Gamon et al., 2005, He et al., 2010a, He et al., 2010b, Avramidis, 2011, Okita and van Genabith, 2011, Callison-Burch et al., 2012].

Experiment #1: NIST Data

In order to assess the performance of the our system combination methodology, we conduct several experiments and measure the translation quality of the resulting hybrid translation output. Note that in the data sets used for

[4] We use constraint decoding as implemented by a customised version of the Moses SMT decoder to compute *comparable* probability scores for all candidate systems, regardless of their internal scores. This allows for a fair comparison between systems.

experimentation individual system names are anonymised as the translation output is part of a shared translation task. We work on training data from the NIST OpenMT12 system combination task, using a held out part of the training data set as reference to evaluate translation quality.

We train binary classification models using libSVM for two language pairs: Arabic→English and Chinese→English. For the first pair we work on translation output generated by $n = 10$ different systems, for the latter pair there are $n = 15$ systems to consider. Note that these numbers differ from the total number of systems per language pair as some systems chose not to be part of the system combination task. The source text originates from the news domain.

We apply our order relation on the given translations to determine a system ranking on the sentence level, similar to what Algorithm 3 describes. Using this information, we then compute pairwise system comparisons as target class labels and annotate individual translations with parser output and language model scores. We use the Stanford Parser [Green and Manning, 2010, Klein and Manning, 2003, Levy and Manning, 2003] to parse both the source text and the corresponding translations. For language model scoring, we use the SRILM toolkit [Stolcke, 2002] training a 5-gram target language model on English Gigaword data. We do not consider source language language models in this work.

Experiment #2: ML4HMT 2012

Datasets

The organisers of the ML4HMT-12 shared task provide two data sets, among these one for the language pair Spanish→English which is the focus in our submission. Participants are given a development bilingual data set aligned

at a sentence level. Each "bilingual sentence" contains:

1. the source sentence;
2. the target (reference) sentence; and
3. the corresponding translations from four individual component MT systems, based on different machine translation paradigms. Two rule-based systems, Apertium as described in [Forcada et al., 2011] and Lucy Translator [Alonso and Thurmair, 2003], and two different variants of Moses [Koehn et al., 2007], a standard phrase-based SMT decoder and a hierarchical, phrase-based system.

The output has been automatically annotated with system-internal meta-data information derived from the translation process of each of the systems. In total, with the development data we receive 20,000 translations per system for training. The test set contains 3,003 sentences and is taken from WMT 2011 ("newstest2011"). Our system competes against five other shared task submissions for the Spanish→English translation task; we will describe these systems below:

DCU-Alignment This submission [Wu et al., 2012] incorporates alignment information as additional meta-data into their system combination module which does not originally utilise any alignment information provided by the individual MT systems producing the candidate translations. The authors add alignment information provided by one of the MT systems, the Lucy Translator rule-based MT engine, into the internal, monolingual, alignment process.

DCU-QE1 This system [Okita et al., 2012a] incorporates a sentence-level Quality Estimation (QE) score as meta-data into their system combination

module. It measures the quality of translations without references. The core idea is to incorporate this knowledge into the system combination module through an improved backbone selection.

DCU-QE2

The third submission [Okita et al., 2012a] also uses a sentence-level Quality Estimation score to do the data selection process. The combined output tends to preserve the translation quality as is expected, which results in a high Meteor score. This approach is comparable to the research conducted in this thesis.

DCU-DA

This system [Okita et al., 2012b] is based on unsupervised topic/genre classification results as meta-data, feeding into the system combination module. Since this module has access to topic/genre information, an MT system can take advantage of this information.

DCU-LM

This submission incorporates latent variables as meta-data into the system combination module. Information about those latent variables are supplied by a probabilistic neural language model.

DFKI

Finally, our submission [Federmann, 2012e] implements a method for system combination based on joint, comparison-based binarisation of feature vectors. It can be used to combine several black-box source systems.

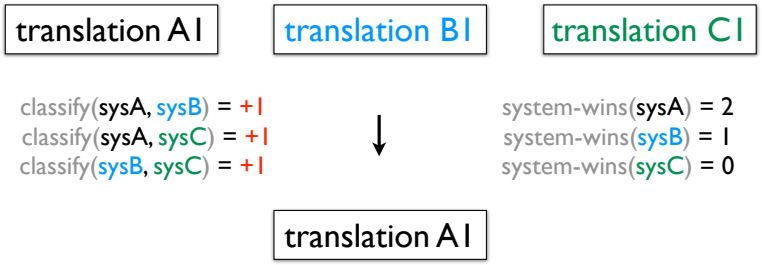


Figure 5.3: Unique winner by majority decision. We simply copy the winning option the final translation output.

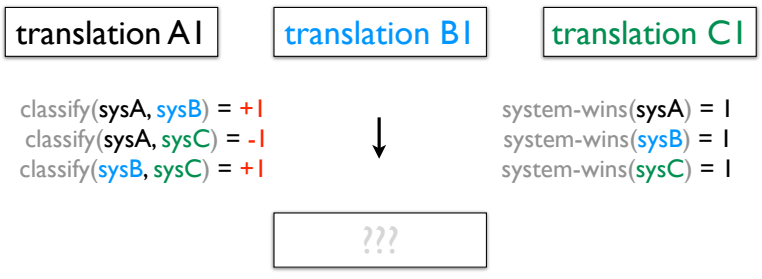


Figure 5.4: Multiple systems could have won. Some conflict resolution or fallback strategy needs to be applied in order to find the best system.

5.5 Results

Experiment #1: NIST Data

Table 5.1 presents results for language pair Arabic→English taken from our experiments with the NIST training data set from OpenMT12. As previously mentioned, we used a held out part of the training data to act as reference in our experiments. We can observe that our combination approach works very well and is able to outperform the single-best system #1. The latter had a best NIST score of 10.1578 while our method achieves 10.3584, a relative increase of +1.97% in terms of NIST score. The same holds true when considering second evaluation metric BLEU. Again, our approach outperforms the single-best BLEU score from system #1, 0.4523 versus 0.4300, a large jump by +5.19%. Overall, it seems that sentence selection functioned nicely with this data set. This might be related to the fact that most of the source systems performed well for both evaluation metrics. Hence, the effect of misclassification is not as critical as for data sets which include candidate translations with more extreme differences.

Table 5.2 gives results for language pair Chinese→English. This time we observe a different performance of our method. By contrast to the previous language pair, sentence selection achieves mixed results, a mediocre 8th rank in terms of NIST score at 7.7636, a decrease of −10.76% compared to the single-best score of system #1, and a slightly better 5th rank in terms of BLEU score, 0.2663 decreasing by −12.52% compared to the top value. It seems that our feature vectors did not allow to discriminate well enough between good and bad systems. Considering that this data set contains 1) more systems than in the previous experiments which 2) also are more diverse in terms of translation quality, the prediction accuracy of our model did not suffice.

Arabic→English		
System	NIST Score	BLEU Score
system #1	10.1578	0.4300
system #2	10.0379	0.4251
system #3	9.8845	0.4179
system #4	9.8841	0.4132
system #5	9.8675	0.4109
system #6	9.8408	0.4070
system #7	9.7120	0.4012
system #8	9.6853	0.3996
system #9	9.6417	0.3982
system #10	9.4226	0.3799
system #11	8.5721	0.3160
system #12	8.1091	0.2746
SVM-combo	1st **10.3584**	1st **0.4523**

Table 5.1: Translation quality measured using NIST and BLEU scores for language pair Arabic→English. Note how our *SVM-combo* system is able to outperform the individual baseline systems for both metrics.

Chinese→English		
System	NIST Score	BLEU Score
system #1	**8.6996**	**0.3044**
system #2	8.4245	0.2927
system #3	8.1160	0.2813
system #4	8.0534	0.2795
system #5	7.9788	0.2587
system #6	7.8969	0.2545
system #7	7.7679	0.2518
system #8	7.6965	0.2369
system #9	7.6461	0.2489
system #10	7.5181	0.2265
system #11	7.4819	0.2580
system #12	7.4045	0.2276
system #13	7.2969	0.2472
system #14	6.8456	0.1957
system #15	6.2852	0.1497
system #16	6.1679	0.1867
SVM-combo	8th 7.7636	5th 0.2663

Table 5.2: Translation quality measured using NIST and BLEU scores for language pair Chinese→English. Here, our *SVM-combo* system only achieves $8th$ rank in terms of NIST score, $5th$ rank according to the BLEU metric.

	Spanish→English					
Score	DCU-Alignment	DCU-QE2	DCU-DA	DCU-LM	DCU-QE1	DFKI
Meteor	0.30692	0.32226	0.32124	0.31684	0.31712	**0.32303**
NIST	7.4296	7.4291	**7.6771**	7.5642	7.6481	7.2830
BLEU	0.2614	0.2524	**0.2634**	0.2562	0.2587	0.2570

Table 5.3: Translation quality of ML4HMT-12 submissions measured using Meteor, NIST, and BLEU scores for language pair Spanish→English.

Figures 5.5 and 5.6 give a graphical overview on experimental results for language pair Arabic→English showing NIST and BLEU scores, respectively. Figures 5.7 and 5.8 present the same information for second language pair Chinese→English. The green line represents the single-best score for each of the automatic metrics, the red line denotes the single-worst such score.

Experiment #2: ML4HMT 2012

Similar to the first edition of the ML4HMT shared task (ML4HMT-11), all submissions to the shared task are evaluated using three automatic scoring metrics, i.e., Meteor [Denkowski and Lavie, 2011], NIST [Doddington, 2002], and BLEU [Papineni et al., 2002], which are all well-renowned evaluation metrics commonly used for MT evaluation. Table 5.3 summarises the results for all participating systems. Our approach, labeled **DFKI**, achieves best overall performance in terms of Meteor, with a value of 0.323. DCU-QE2 and DCU-DA are close, though. This supports our Hypothesis 6, *The performance of our system combination approach is competitive compared to other methods.* We have meanwhile experimented with multiple binary classification models, up to one model per pairwise comparison A, B: on the same ML4HMT-12 data, such a more specialised suite of classifiers allows to increase to 0.336. While still being far away from the theoretical upper bound (0.367) it represents a +4.67% increase (relative) compared to other systems.

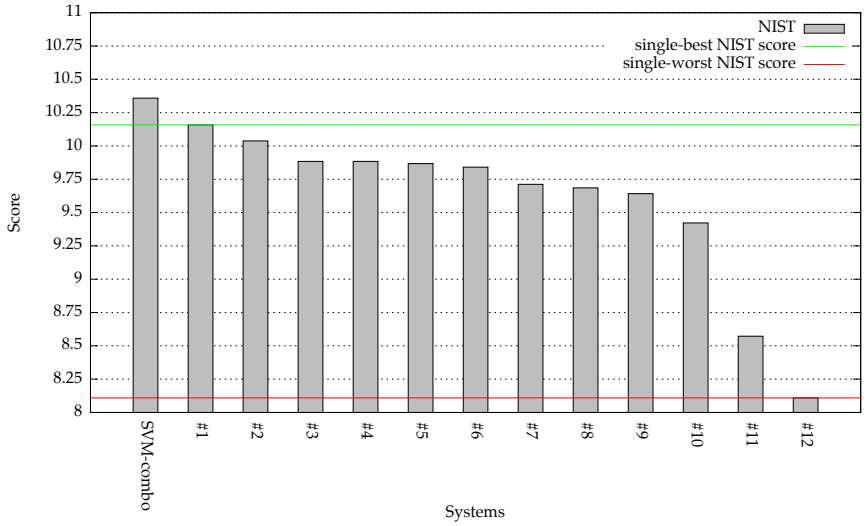


Figure 5.5: NIST Scores for language pair Arabic→English. *SVM-combo* shows the performance of our sentence selection approach for hybrid MT. Note how we are able to outperform the single-best system #1.

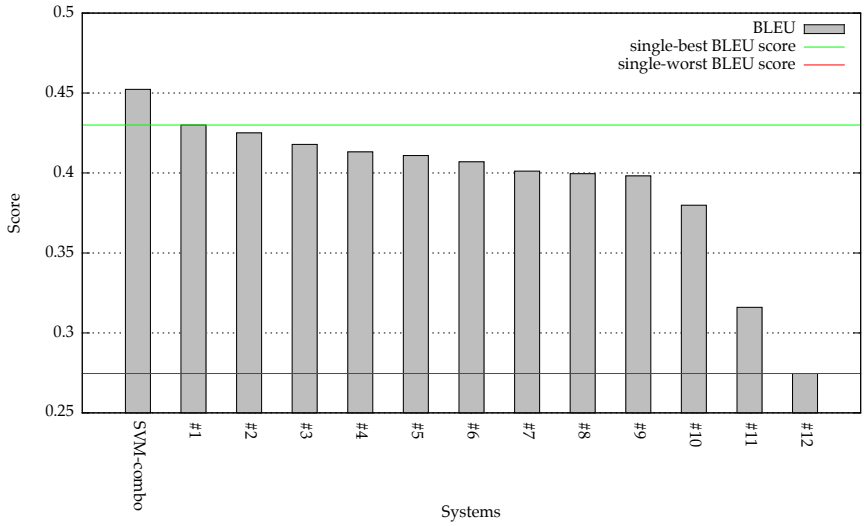


Figure 5.6: BLEU Scores for language pair Arabic→English. As above, *SVM-combo* presents the performance of our method. Again, we are able to outperform the single-best system #1, achieving 0.45 instead of 0.43.

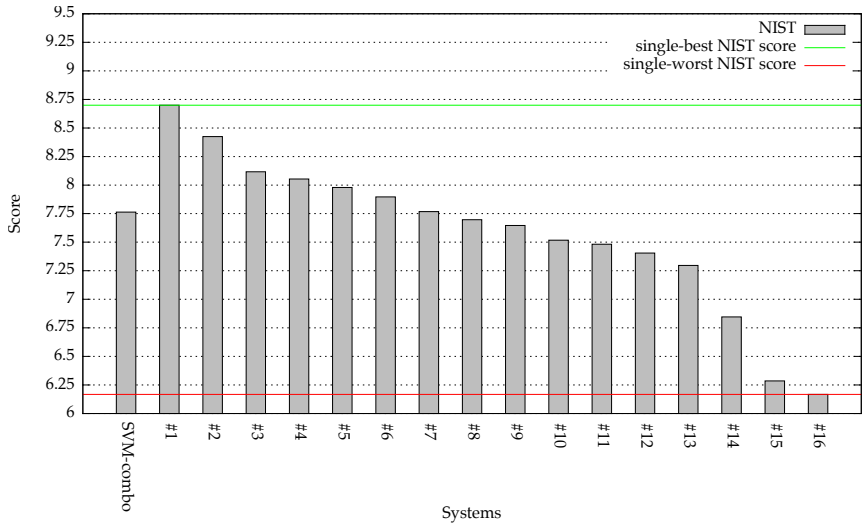


Figure 5.7: NIST Scores for language pair Chinese→English. *SVM-combo* shows the performance of our classification-based method for system combination. By contrast to language pair Arabic→English, we are not able outperform the single-best system #1. In fact we only achieve 8th rank.

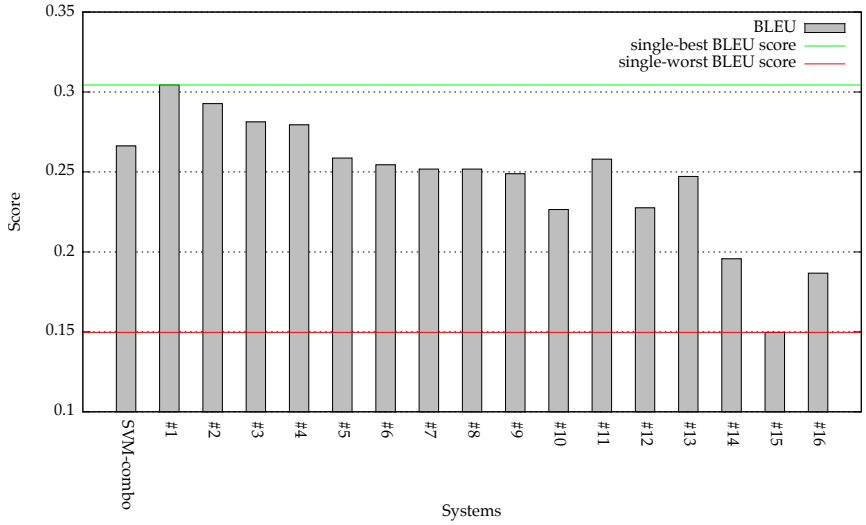
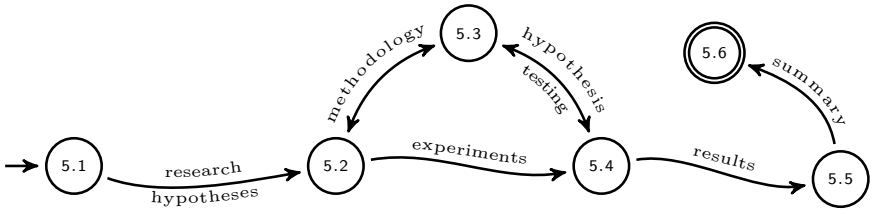


Figure 5.8: BLEU Scores for language pair Chinese→English. Similar to above, *SVM-combo* represents our sentence selection approach. Again, we are not able to outperform the single-best system #1, only reaching rank #5.

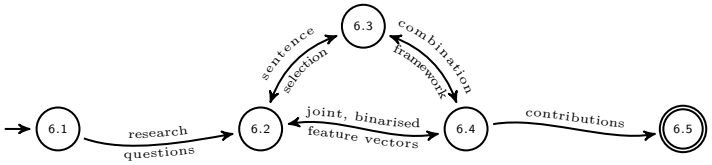


5.6 Chapter Summary

In this chapter we have combined the idea of system combination using a sentence selection mechanism and the novel notion of joint, comparison-based binarisation of feature vectors into a framework for hybrid machine translation. We have stated our research hypothesis and carefully verified it. Hypothesis 6,

> *It is possible to define and implement a competitive sentence selection approach for system combination which is based on binary classification models trained using joint, comparison-based binarisation of feature vectors.*

holds as we have been able to implement our framework for the experiments in this chapter and as we have been able to observe competitive performance compared to competing baseline or combination systems in our experiments. In essence, we have been successful in our attempt to define and implement a system combination method based on binary classification estimated using joint, comparison-based binarisation of feature vectors.



6 Conclusion

"I never think of the future. It comes soon enough."

– **Albert Einstein**: interview given on the *Belgenland*, December 1930.

"We can be heroes
Just for one day."

– **David Bowie**: *Heroes*, 1997.

"New opinions are always suspected, and usually opposed,
without any other reason but because they are not already common."

– **John Locke**: *An Essay Concerning Human Understanding*, 1690.

In this thesis, we have described the design and implementation of a system combination framework using binary classification models which have been estimated using joint, comparison-based binarisation of feature vectors. We have first presented oracle experiments to empirically establish upper bounds for sentence selection in Chapter 3. Afterwards, we have introduced and evaluated the idea of joint, comparison-based binarisation of feature vectors in Chapter 4. In Chapter 5, we have presented the resulting system combination framework. In this chapter, we provide a summary of our findings, comparing our research hypotheses to the results from our experiments, and we give an outlook to future research topics which result from this thesis.

6.1 Performance of Sentence Selection

Sentence selection is a main component of our system combination framework. By contrast to confusion network decoding, we do not divide given candidate translations into n-grams, i.e., sub-segments below the sentence level which are then recombined to form a (hopefully) improved translation. Instead, we leave segments/sentences intact and choose one of the available candidates for each individual segment, in unaltered form: *e pluribus unum, immutatum*.

Our first research hypothesis states that sentence selection methods can theoretically result in an improved overall translation quality, compared to the quality of the individual source systems from which we synthesise the combined translation.

Hypothesis 1. *Sentence selection methods can outperform their source systems on real data.*

We have verified this hypothesis by conducting an extensive experiment on data from the yearly Workshop on Statistical Machine Translation (WMT). Our research considers translations from 2007 until 2013, thus being the largest meta-study on official WMT results so far. Based on our findings, we can confirm Hypothesis 1; empirically, sentence selection methods are well able to outperform their individual source machine translation systems.

Next, we want to find out if empirical performance gains of selection-based methods for system combination can be observed across the technological paradigms of the given candidate systems or the language pair under investigation.

Hypothesis 2. *Sentence selection approaches show improvement potential across language pairs or underlying technological paradigms of the source systems.*

We are able to verify this hypothesis in our experiments. Theoretical performance gains achievable by optimal sentence selection are not related to the language pair or the methodology of the source systems. For the latter finding (which has not been discussed in Chapter 3), see [Federmann, 2012d] which presents the corresponding experiments.

A final matter of investigation is the contribution of bad candidate translations. We want to find out whether translation systems which perform bad on the overall system level can be beneficial in the context of sentence selection. We assume that they can likely contribute something to our final combination results. Our final working hypothesis in Chapter 3 is:

Hypothesis 3. *Even systems which perform bad on the global system level can contribute helpful segment translations.*

We find evidence supporting this hypothesis. Sometimes, even the combination of the k-worst candidate translations can outperform the single-best source system. This implies that even globally bad systems do, at times, contain information *nuggets* to mine for.

6.2 Joint, Comparison-Based Binarisation of Features

The second fundamental contribution of this thesis is the introduction of joint, comparison-based binarisation of feature vectors. These explicitly model comparison of two systems A, B and, hence, are more effective for training binary classification models which are used for such comparison operations.

Before considering the binarisation step, however, we investigate the performance of joint feature vectors by comparison to the corresponding single feature vectors. The corresponding research hypothesis reads as follows:

Hypothesis 4. *Explicit modelling of pairwise system comparison using what we call joint feature vectors can outperform single feature vectors in terms of both resulting prediction accuracy and faster training times for large data sets.*

In our experiments, we observe that joint feature vectors have an edge on their single feature vector counterparts. This dwindles, however, with increasing amounts of random noisy features. Even for these, joint feature vectors are faster to train which verifies our working hypothesis.

The binarisation of joint feature vectors based on the comparison of individual feature values further optimises them for binary comparison of systems. Our evidence shows that joint, comparison-based binarisation of feature vectors can outperform their joint counterparts. For joint feature vectors which are composed of feature values for two candidate systems, the machine learning approach which is used has to guess which of the feature values correspond to each other and actually describe the same property of the two translations. Joint, comparison-based feature binarisation explicitly encodes such relationships. Thus, our second research hypothesis in Chapter 4 is:

Hypothesis 5. *Binarisation of the comparison results obtained by comparing the individual feature values from two systems A, B can outperform the corresponding joint feature vectors.*

We compare joint feature vectors and joint, comparison-based binarisation of feature vectors based on so-called ROC curves. These plot the true

positive rate (TPR) against the false positive rate (FPR), at various threshold parameters which control the estimation process of the classification model under investigation. Performance of joint, comparison-based binarisation of feature vectors is much better than that of the corresponding joint feature vectors. This even holds for large numbers of noisy features, which confirms our hypothesis.

6.3 System Combination Framework

Based on the successful verification of our research hypotheses regarding 1) the potential performance of sentence selection for system combination, and 2) the effectiveness of joint, comparison-based binarisation of feature vectors, we propose a framework for system combination in Chapter 5. This effectively combines results from Chapters 3 and 4. In essence, we are trying to verify that the fundamental findings from these chapters can be turned into a competitive combination approach.

Hypothesis 6. *It is possible to define and implement a sentence selection approach for system combination which is based on binary classification models trained with joint, comparison-based binarisation of feature vectors that is competitive with state-of-the-art system combination methods.*

The definition and implementation of such a framework is the main contribution of Chapter 5. Our final research hypothesis aims at the comparison of our methodology and state-of-the-art methods based on confusion networks. These have been most actively investigated in the last decade and, thus, are a baseline we have to compare our method against in order to rightfully claim that a sentence selection approach trained using joint, comparison-based binarisation of feature vectors is competitive.

Based on the findings from our experiments, we are able to confirm this hypothesis. System combination based on binary classification models which have been estimated using joint, comparison-based binarisation of feature vectors can outperform other state-of-the-art methods. We have observed this 1) for our experiments on NIST OpenMT12 data, and 2) for the ML4HMT 2012 shared task where our implementation performed best in terms of Meteor scores.

In summary, we have defined and implemented a novel method for system combination in the context of machine translation. It is based on sentence selection following the decisions of one or several binary classification models. The classifiers are estimated using a joint, binarised feature combination approach which explicitly models comparison of systems A, B on the level of feature vectors.

We are able to observe improvements of both sentence selection and binarisation of feature vectors on a theoretical level. Also, our method achieves competitive performance in experiments on real data. Thus, we are able to confirm Hypothesis 6.

Limitations of our Work

In the work conducted for this thesis, we have investigated whether a sentence selection method for system combination of machine translation output can achieve competitive performance compared to state-of-the-art approaches based on confusion networks. Our results show that the Meteor metric, which provides usable scores on the sentence level, can (in combination with NIST and BLEU for increased robustness in case of equality under Meteor) be used to achieve this. Other automatic metrics which produce reliable scores for individual segments can theoretically also be applied instead of Meteor.

Combinations are also possible. We leave this area for future research.

Furthermore, we have not provided an exhaustive search for linguistic features which can be used with this method. It is important to note that our concept of joint, comparison-based binarisation of feature values can be used with *any* feature value which produces *comparable* output values. This means that most features which can be interpreted in isolation by human annotators are, in fact, usable with our approach. More detailed experiments which compare the performance of our approach to other system combination and machine learning methods using a fixed set of shared (and, thus, comparable) feature values are required to conclusively show that our method is better. While our empirical evidence suggests that our approach has competitive performance, additional research is needed to formally prove this. Again, we leave these experiments to future work.

6.4 Future Research Work

In the future several extensions of the research work described in this thesis are possible. We discuss three interesting ways of continuation. These are 1) investigation of features for improved system combination, 2) application of our selection methodology on the level of phrases, or 3) combination with quality estimation as input feature. All three may allow to further refine our system combination approach.

Better Features for System Combination

In our research work, we have used a selection of linguistic features which evaluate characteristic properties of candidate translations. Such features include language model scores, lexical phrase table probabilities, or feature values derived from parsing source text and candidate translations. While we

have identified several useful features during the course of our experiments, an in-depth inspection of these has not been the focus of our research. A more focused and detailed investigation can likely result in more profound findings in this respect. Machine learning tools can be used in combination with methods for *feature selection* to determine an optimal subset of features for the classification task at hand.

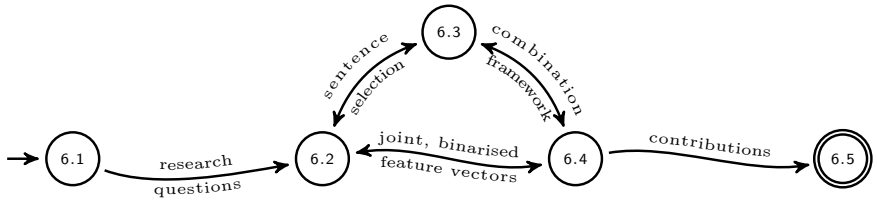
Going to Phrase Level

In our motivation in Chapter 1, we have stated that any sentence selection approach has the drawback of not being able to integrate good phrases from different candidate translations. This is due to the very design of sentence selection which does not alter the selection translation in any way. Of course, it would be interesting to more closely investigate how a selection-based system combination approach as ours can be extended to perform selection on the level of phrases. We have previously worked on a methodologically similar method for hybrid machine translation, see [Federmann et al., 2009]. Using one of the candidate translations as output template we can compute phrase alignments to the other candidates. Using such alignment links it seems possible to combine in good phrases from several systems. Of course, output quality heavily depends on the quality of the alignment process. Furthermore, we lose the guarantee w.r.t. preservation of syntactic and semantic features of the resulting, combined translation. The effects of such a modification, hence, are an interesting research topic.

Integrating Quality Estimation

Finally, it seems a good idea to combine the system combination framework presented in this thesis with current results from the field of quality estima-

tion. Such methods aim at estimating the quality of a given translation without having access to the reference. This is exactly the application scenario we face when applying our combination methodology. Instead of using linguistic feature for our joint, comparison-based binarisation of feature vectors which feed into our binary classification models, we can use non-linguistic feature values derived from one or several quality estimation models. Depending on the discriminative capabilities of the individual estimators, this meta-combination should work nicely. In fact, it would allow us to benefit from advances made in the research field of quality estimation and potentially reduce our dependency on more and better linguistic features.



After having summarised our findings and given an outlook on future work, we conclude with a brief summary of the major contributions we have achieved in our thesis research.

6.5 Thesis Contributions

The major contributions of this thesis are:

- We have conducted the **largest meta-study on WMT results** published by the Workshop on Statistical Machine Translation (2007–2013);
- The definition of **joint, comparison-based binarisation of features** for machine learning of binary classification between several candidate translations;
- We have shown that **sentence selection approaches can perform** at the same level of translation quality as existing, state-of-the-art system combination methods;
- Our evaluation software for assessment of machine translation quality, **Appraise**, has become the official evaluation system of WMT 2013.

Appendices

A

Data for Optimal Sentence Selection

We have conducted the largest meta-study on data collected by the yearly Workshops on Statistical Machine Translation (WMT). We have defined and investigated different combinations of the given set of candidate systems, implementing three different combination strategies, namely Top-k, Worst-k, and Alternate-k. For each of these combination sets, we then computed $\text{Meteor}_{\text{AVG}}$ scores simulating optimal sentence selection to measure what performance gain could be attained. All data which has been collected and aggregated during our experiments is available from the author's GitHub repository at `http://www.github.com/cfedermann/`.

B

Data for Binarisation Experiments

Table B.1 gives all results from our comparison of joint feature vectors and their joint, binarised counterparts. Both are estimated using `svm.LinearSVC` and trained for subset size 2,000–12,000 with noise level increasing from 0 up to 10,000. Note the superior performance of joint, binarised feature vectors, especially for large amounts of noise.

Likewise, Table B.2 gives results from our comparison of joint feature vectors and their joint, binarised counterparts. By contrast, classification models are estimated using `svm.SVC` and trained for subset size 2,000–12,000 with noise level increasing from 0 up to 10,000. While joint feature vectors see an improvement in terms of prediction accuracy, they are still outperformed by joint, binarised feature vectors.

Finally, Table B.3 presents results from our comparison of joint feature vectors and their joint, binarised counterparts estimated with `svm.SVC` using an rbf kernel. Again, we train models for subset size 2,000–12,000 with noise level increasing from 0 up to 10,000. Similar to our previous findings, the performance of joint, binarised feature vectors is superior.

	Joint feature vectors					Joint, binarised feature vectors				
Noise level	2,000	4,000	6,000	9,000	12,000	2,000	4,000	6,000	9,000	12,000
0	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
10	0.97	0.97	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
20	0.97	0.97	0.98	0.98	0.99	0.99	0.99	0.99	0.99	0.99
50	0.96	0.97	0.98	0.98	0.98	0.99	0.99	0.99	0.99	0.99
100	0.93	0.96	0.97	0.98	0.98	0.97	0.99	0.99	0.99	0.99
200	0.92	0.95	0.96	0.97	0.97	0.98	0.99	0.99	0.99	0.99
500	0.80	0.90	0.93	0.95	0.96	0.99	0.99	0.99	0.99	0.99
1,000	0.57	0.80	0.88	0.92	0.93	0.99	0.99	0.99	0.99	0.99
5,000	0.51	0.58	0.56	0.58	0.58	0.83	0.94	0.98	0.99	0.99
10,000	0.53	0.58	0.58	0.57	0.56	0.73	0.83	0.89	0.95	0.98

Table B.1: **Subset: 2,000–12,000, Kernel: linear, Type: LinearSVC**
Comparison of joint and joint binarised feature vectors

	Joint feature vectors					Joint, binarised feature vectors				
Noise level	2,000	4,000	6,000	9,000	12,000	2,000	4,000	6,000	9,000	12,000
0	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
10	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
20	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
50	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
100	0.97	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
200	0.96	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
500	0.85	0.95	0.97	0.98	0.99	0.99	0.99	0.99	0.99	0.99
1,000	0.60	0.88	0.94	0.97	0.98	0.99	0.99	0.99	0.99	0.99
5,000	0.49	0.62	0.59	0.62	0.62	0.91	0.98	0.99	0.99	0.99
10,000	0.53	0.61	0.60	0.59	0.58	0.83	0.92	0.96	0.99	0.99

Table B.2: **Subset: 2,000–12,000, Kernel: linear, Type: SVC**
Comparison of joint and joint binarised feature vectors

	Joint feature vectors					Joint, binarised feature vectors				
Noise level	2,000	4,000	6,000	9,000	12,000	2,000	4,000	6,000	9,000	12,000
0	0.93	0.95	0.97	0.98	0.99	0.99	0.99	0.99	0.99	0.99
10	0.81	0.88	0.91	0.93	0.96	0.99	0.99	0.99	0.99	0.99
20	0.75	0.85	0.88	0.89	0.92	0.99	0.99	0.99	0.99	0.99
50	0.71	0.81	0.85	0.84	0.85	0.99	0.99	0.99	0.99	0.99
100	0.68	0.79	0.82	0.81	0.81	0.97	0.99	0.99	0.99	0.99
200	0.66	0.77	0.80	0.78	0.77	0.98	0.99	0.99	0.99	0.99
500	0.61	0.73	0.77	0.75	0.73	0.99	0.99	0.99	0.99	0.99
1,000	0.61	0.73	0.75	0.72	0.70	0.99	0.99	0.99	0.99	0.99
5,000	0.50	0.68	0.70	0.69	0.64	0.89	0.95	0.98	0.99	0.99
10,000	0.50	0.65	0.67	0.66	0.61	0.81	0.90	0.94	0.96	0.98

Table B.3: **Subset: 2,000–12,000, Kernel: rbf, Type: SVC**
Comparison of joint and joint binarised feature vectors

C

Appraise

Evaluation of Machine Translation output to assess translation quality is a difficult task. Automatic metrics such as BLEU [Papineni et al., 2002] or Meteor [Denkowski and Lavie, 2011] are commonly used in minimum error rate training [Och, 2003] for tuning of MT systems. Also, they are used as evaluation metrics. The main problem in designing automatic quality metrics for MT is to achieve a high correlation with human judgments on the same translation output. While current metrics show promising performance in this respect, manual inspection and evaluation of MT results is still equally important. The manual analysis of a given, machine translated text is a time-consuming and laborious process; it involves training of annotators, requires detailed and clear-cut annotation guidelines, and—last but not least—an annotation software that allows annotators to get their job done quickly and efficiently.

In this chapter, we describe *Appraise*, an open-source tool that allows to perform manual evaluation of Machine Translation output. Appraise can be used to collect human judgments (*or any other annotation*) on translation output, implementing several annotation tasks. We will describe the tool in more detail on the following pages.[1]

[1] This chapter is based on [Federmann, 2012b], with updates regarding WMT 2013.

Motivation

As we have mentioned before, the collection of manual judgments on MT output is a tedious task; this holds for simple tasks such as translation ranking but also for more complex challenges like word-level error analysis or post-editing of translation output. Annotators tend to lose focus after several sentences, resulting in reduced intra-annotator agreement and increased annotation time. In our experience with manual evaluation campaigns it has shown that a well-designed annotation tool helps to overcome such issues.

Development of the Appraise software package started back in 2009 as part of the EuroMatrixPlus project where the tool was used to quickly compare different sets of candidate translations from our hybrid machine translation engine to get an indication whether our system improved or degraded in terms of translation quality. A first version of Appraise was released and described by [Federmann, 2010].

System Description

In a nutshell, Appraise is an open-source tool for manual evaluation and annotation of machine translation output. It allows to collect human judgments on given translation output, implementing annotation tasks such as (but not limited to):

- translation quality checking;
- ranking of translations;
- error classification;
- manual post-editing.

We will provide a more detailed discussion of these tasks in Section C.

The software features an extensible XML import/output format and can easily be adapted to new annotation tasks. The software also features support for the automatic computation of inter-annotator agreement scores, allowing quick access to evaluation results. We currently support computation of the following inter-annotator agreement scores:

- Krippendorff's α as described by [Krippendorff, 2004];
- Fleiss' κ as published in [Fleiss, 1971], extending [Cohen, 1960];
- Bennett, Alpert, and Goldsteins S as defined in [Bennett et al., 1954];
- Scott's π as introduced in [Scott, 1955].

Agreement computation is based on code which has been implemented by the NLTK project [Bird et al., 2009]. Additional agreement metrics can be added easily—in fact, we implemented our own version of κ during the WMT 2013 evaluation campaign to maintain compatibility with previous editions of the shared task; the visualisation of agreement scores or other annotation results can be adapted to best match the corresponding task design.

Appraise has been implemented using the Python-based *Django web framework*[2] which takes care of low-level tasks such as "HTTP handling", database modeling, and object-relational mapping. We use *Bootstrap*[3] as basis for the design of the application and implemented it using long-standing and well-established open-source software with large communities supporting them in the hope that this will also benefit the Appraise software package.

We have opened up Appraise development and released the source code on GitHub at `https://github.com/cfedermann/Appraise`. Anybody with a free GitHub account may fork the project and create an own version of the software. Due to the flexibility of the `git` source code management system,

[2] See `http://www.djangoproject.com/` for more information
[3] Available from `http://getbootstrap.com/`

it is easy to re-integrate external changes into the master repository, allowing other developers to feed back bug fixes and new features, thus improving and extending the original software. Appraise is available under an open, BSD-style license.[4]

Annotation Tasks

We have developed several annotation tasks which are useful for evaluation of machine translation output. The following task types are implemented in the GitHub version of Appraise:

1. **Ranking** The annotator is shown 1) the source sentence and 2) a set of several ($n \geq 2$) candidate translations. It is also possible to additionally present the reference translation. Wherever available, one sentence of left/right context is displayed to support the annotator in the process. We also have implemented a special *3-way ranking task* which works for pairs of candidate translations and gives the annotator an intuitive interface for quick $A > B$, $A = B$, or $A < B$ classification.

2. **Error Classification** The annotator is presented 1) the source sentence and 2) a candidate translation which has to be inspected with respect to errors contained in the translation output. We use a refined version of the classification described in [Vilar et al., 2006]. Error annotation is possible on the sentence level as well as for individual words. The annotator can choose to skip translations containing "too many errors" and is able to differentiate between "minor" and "severe" errors.

[4] See `https://raw.github.com/cfedermann/Appraise/master/appraise/LICENSE`

3. **Quality Estimation** The annotator is given 1) the source sentence and 2) one candidate translation which has to be classified as *Acceptable*, *Can easily be fixed*, or *None of both*. We also show the reference sentence and again present left/right context if available. This task can be used to get a quick estimate on the *acceptability* of a set of translations.

4. **Post-editing** The annotator is shown 1) the source sentence including left/right context wherever available and 2) one or several candidate translation. The task is defined as choosing the translation which is "easiest to post-edit" and then performing the post-editing operation on the selected translation.

   In some of our experiments with Appraise, we found that annotators did not necessarily choose the overall best candidate translation for post-editing but often selected worse translations which, however, could be post-edited more quickly.

Changes for WMT 2013

In order to use Appraise for the human evaluation campaign conducted as part of the shared translation task at WMT 2013, we updated the toolkit in several ways. We added a new ranking mode which more closely resembled what previous WMT evaluation campaigns had implemented and connected the Appraise evaluation toolkit to Amazon's Mechanical Turk framework for crowd-based annotation. Results from the WMT 2013 manual evaluation campaign are discussed in the official results paper by [Bojar et al., 2013]. An extension to other crowd-sourcing frameworks is planned and estimated to be finished in time for WMT 2014.

Bibliography

[Alonso and Thurmair, 2003] Alonso, J. A. and Thurmair, G. (2003). The Comprendium Translator system. In *Proceedings of the Ninth Machine Translation Summit*, New Orleans, USA.

[Avramidis, 2011] Avramidis, E. (2011). DFKI System Combination with Sentence Ranking at ML4HMT-2011. In *Proceedings of the International Workshop on Using Linguistic Information for Hybrid Machine Translation (LIHMT 2011) and of the Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT)*, Barcelona, Spain.

[Avramidis et al., 2012] Avramidis, E., Costa-Jussà, M. R., Federmann, C., van Genabith, J., Melero, M., and Pecina, P. (2012). A Richly Annotated, Multilingual Parallel Corpus for Hybrid Machine Translation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 2189–2193, Istanbul, Turkey. European Language Resources Association (ELRA).

[Aziz et al., 2012] Aziz, W., de Sousa, S. C. M., and Specia, L. (2012). PET: a Tool for Post-editing and Assessing Machine Translation. In *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC '12, pages 3982–3987, Istanbul, Turkey. European Language Resources Association (ELRA).

[Bennett et al., 1954] Bennett, E. M., Alpert, R., and Goldstein, A. C. (1954). Communications Through Limited-response Questioning. *Public Opinion Quarterly*, 18(3):303–308.

[Bird et al., 2009] Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly, Beijing.

[Bojar et al., 2013] Bojar, O., Buck, C., Callison-Burch, C., Federmann, C., Haddow, B., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2013). Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, WMT '13, pages 1–44, Sofia, Bulgaria. Association for Computational Linguistics.

[Brown et al., 1993] Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

[Callison-Burch, 2007] Callison-Burch, C. (2007). *Paraphrasing and Translation*. PhD thesis, University of Edinburgh, Edinburgh, Scotland, UK.

[Callison-Burch et al., 2007] Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2007). (meta-) Evaluation of Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, WMT '07, pages 136–158, Prague, Czech Republic. Association for Computational Linguistics.

[Callison-Burch et al., 2008] Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2008). Further Meta-Evaluation of Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, WMT '08, pages 70–106, Columbus, Ohio, USA. Association for Computational Linguistics.

[Callison-Burch et al., 2010] Callison-Burch, C., Koehn, P., Monz, C., Peterson, K., Przybocki, M., and Zaidan, O. (2010). Findings of the 2010 Joint Workshop on Statistical Machine Translation and Metrics for Machine Translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, WMT '10, pages 17–53, Uppsala, Sweden. Association for Computational Linguistics. Revised August 2010.

[Callison-Burch et al., 2012] Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2012). Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 10–51, Montréal, Canada. Association for Computational Linguistics.

[Callison-Burch et al., 2009] Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Ma-*

chine Translation, WMT '09, pages 1–28, Athens, Greece. Association for Computational Linguistics.

[Callison-Burch et al., 2011] Callison-Burch, C., Koehn, P., Monz, C., and Zaidan, O. (2011). Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 22–64, Edinburgh, Scotland, UK. Association for Computational Linguistics.

[Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[Chiang, 2007] Chiang, D. (2007). Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

[Chomsky, 1965] Chomsky, N. (1965). *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA, USA, 1st edition.

[Cohen, 1960] Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46.

[de Saint-Exupéry, 1943] de Saint-Exupéry, A. (1943). *Le Petit Prince*. Reynal & Hitchcock, New York, NY, USA.

[Denkowski and Lavie, 2011] Denkowski, M. and Lavie, A. (2011). Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 85–91, Edinburgh, Scotland, UK. Association for Computational Linguistics.

[Doddington, 2002] Doddington, G. (2002). Automatic Evaluation of Machine Translation Quality Using n-gram Co-occurrence Statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, HLT '02, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Dreyer and Marcu, 2012] Dreyer, M. and Marcu, D. (2012). HyTER: Meaning-Equivalent Semantics for Translation Evaluation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL '12, pages 162–171, Montréal, Canada. Association for Computational Linguistics.

[Dyer et al., 2010] Dyer, C., Lopez, A., Ganitkevitch, J., Weese, J., Ture, F., Blunsom, P., Setiawan, H., Eidelman, V., and Resnik, P. (2010). cdec: A Decoder, Alignment, and Learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, ACL '10, pages 7–12, Uppsala, Sweden. Association for Computational Linguistics.

[Eisele and Chen, 2010] Eisele, A. and Chen, Y. (2010). MultiUN: A Multilingual Corpus from United Nation Documents. In Tapias, D., Rosner, M., Piperidis, S., Odjik, J., Mariani, J., Maegaard, B., Choukri, K., and Calzolari, N., editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation*, LREC '10, pages 2868–2872, Valetta, Malta. European Language Resources Association (ELRA).

[Fan et al., 2008] Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9(1):1871–1874.

[Federico et al., 2008] Federico, M., Bertoldi, N., and Cettolo, M. (2008). IRSTLM: an Open Source Toolkit for Handling Large Scale Language Models. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association*, INTERSPEECH '08, pages 1618–1621, Brisbane, Australia. International Speech Communication Association.

[Federmann, 2010] Federmann, C. (2010). Appraise: An Open-Source Toolkit for Manual Phrase-Based Evaluation of Translations. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, LREC '10, pages 1731–1734, Valletta, Malta. European Language Resources Association (ELRA).

[Federmann, 2011] Federmann, C. (2011). Results from the ML4HMT Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation. In *Proceedings of the International Workshop on Using Linguistic Information for Hybrid Machine Translation (LIHMT 2011) and of the Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT-11)*, pages 110–117, Barcelona, Spain. META-NET.

[Federmann, 2012a] Federmann, C. (2012a). A Machine-Learning Framework for Hybrid Machine Translation. In *Proceedings of the 35th Annual German Conference on Artificial Intelligence (KI-2012)*, KI '12, pages 37–48, Saarbrücken, Germany. Springer, Heidelberg.

[Federmann, 2012b] Federmann, C. (2012b). Appraise: An Open-Source Toolkit for Manual Evaluation of Machine Translation Output. *The Prague Bulletin of Mathematical Linguistics (PBML)*, 98:25–35.

[Federmann, 2012c] Federmann, C. (2012c). Can Machine Learning Algorithms Improve Phrase Selection in Hybrid Machine Translation? In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics. Joint Workshop on Exploiting Synergies between Information Retrieval and Machine Translation (ESIRMT) and Hybrid Approaches to Machine Translation (HyTra) (ESIRMT-HyTra-12), located at 13th Conference of the European Chapter of the Association for Computational Linguistics*, HyTra '12, pages 113–118, Avignon, France. European Chapter of the Association for Computational Linguistics (EACL).

[Federmann, 2012d] Federmann, C. (2012d). Hybrid Machine Translation Using Joint, Binarised Feature Vectors. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas (AMTA 2012)*, AMTA '12, pages 113–118, San Diego, USA. Association for Machine Translation in the Americas (AMTA).

[Federmann, 2012e] Federmann, C. (2012e). System Combination Using Joint, Binarised Feature Vectors. In *Proceedings of the Second Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT-12)*, Mumbai, India.

[Federmann et al., 2011] Federmann, C., Hunsicker, S., Wolf, P., and Bernardi, U. (2011). D2.3: Study of the integration of stochastic tree models in LT analysis phase and of the integration of bilingual language models in LT transfer phase. Technical report, EuroMatrix Plus consortium, Saarbrücken, Germany.

[Federmann et al., 2012] Federmann, C., Melero, M., Pecina, P., and van Genabith, J. (2012). Towards Optimal Choice Selection for Improved Hybrid Machine Translation. *The Prague Bulletin of Mathematical Linguistics (PBML)*, 97:5–22.

[Federmann et al., 2009] Federmann, C., Theison, S., Eisele, A., Uszkoreit, H., Chen, Y., Jellinghaus, M., and Hunsicker, S. (2009). Translation Combination using Factored Word Substitution. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, WMT '09, pages 70–74, Athens, Greece. Association for Computational Linguistics.

[Fellbaum, 1998] Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, USA, 1st edition.

[Fleiss, 1971] Fleiss, J. (1971). Measuring Nominal Scale Agreement among Many Raters. *Psychological Bulletin*, 76(5):378–382.

[Forcada et al., 2011] Forcada, M. L., Ginestí-Rosell, M., Nordfalk, J., O'Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Sánchez-Martínez, F., Ramírez-Sánchez, G., and Tyers, F. M. (2011). Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25:127–144.

[Frederking and Nirenburg, 1994] Frederking, R. and Nirenburg, S. (1994). Three Heads are Better Than One. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, ANLC '94, pages 95–100, Stuttgart, Germany. Association for Computational Linguistics.

[Gamon et al., 2005] Gamon, M., Aue, A., and Smets, M. (2005). Sentence-level MT Evaluation Without Reference Translations: Beyond Language Modeling. In *Proceedings of the 10th EAMT Conference "Practical applications of machine translation"*, pages 103–111. European Association for Machine Translation.

[Green and Manning, 2010] Green, S. and Manning, C. D. (2010). Better Arabic Parsing: Baselines, Evaluations, and Analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 394–402, Stroudsburg, PA, USA. Association for Computational Linguistics.

[He et al., 2010a] He, Y., Ma, Y., van Genabith, J., and Way, A. (2010a). Bridging SMT and TM with Translation Recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 622–630, Stroudsburg, PA, USA. Association for Computational Linguistics.

[He et al., 2010b] He, Y., Ma, Y., Way, A., and van Genabith, J. (2010b). Integrating N-best SMT Outputs into a TM System. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 374–382, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Hildebrand and Vogel, 2008] Hildebrand, A. S. and Vogel, S. (2008). Combination of Machine Translation Systems via Hypothesis Selection from

Combined N-Best Lists. In *MT at work: Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, AMTA '08, pages 254–261, Waikiki, Hawaii, USA. Association for Machine Translation in the Americas.

[Hildebrand and Vogel, 2009] Hildebrand, A. S. and Vogel, S. (2009). CMU System Combination for WMT'09. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, WMT '09, pages 47–50, Athens, Greece. Association for Computational Linguistics.

[Hildebrand and Vogel, 2010] Hildebrand, A. S. and Vogel, S. (2010). CMU System Combination via Hypothesis Selection for WMT'10. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, WMT '10, pages 307–310, Uppsala, Sweden. Association for Computational Linguistics.

[Hunsicker et al., 2012] Hunsicker, S., Yu, C., and Federmann, C. (2012). Machine Learning for Hybrid Machine Translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 312–316, Montréal, Canada. Association for Computational Linguistics.

[Hutchins, 1986] Hutchins, W. J. (1986). *Machine Translation: Past, Present, Future*. Ellis Horwood Series in Computers and Their Applications. Longman Higher Education, Chichester, UK, 1st edition.

[Hutchins, 2004] Hutchins, W. J. (2004). The Georgetown-IBM Experiment Demonstrated in January 1954. In *Machine Translation: From Real Users to Research*, Lecture Notes in Computer Science, pages 102–114, Berlin, Germany. Springer.

[Kelly and Zetzsche, 2012] Kelly, N. and Zetzsche, J. (2012). *Found in Translation: How Language Shapes Our Lives and Transforms the World*. Perigee Trade, New York, NY, USA, 1st edition.

[Klein and Manning, 2003] Klein, D. and Manning, C. (2003). Accurate Unlexicalized Parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, volume 1 of *ACL '03*, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Koehn, 2005] Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT Summit X: Proceedings of the Tenth Machine Translation Summit*, MT Summit '05, pages 79–86, Phuket, Thailand. Asia-Pacific Association for Machine Translation.

[Koehn, 2010] Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press, Cambridge, UK, 1st edition.

[Koehn and Hoang, 2007] Koehn, P. and Hoang, H. (2007). Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, EMNLP '07, pages 868–876, Prague, Czech Republic. Association for Computational Linguistics.

[Koehn et al., 2007] Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, ACL '07, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

[Krippendorff, 2004] Krippendorff, K. (2004). Reliability in Content Analysis. Some Common Misconceptions and Recommendations. *Human Communication Research*, 30(3):411–433.

[Levenberg et al., 2010] Levenberg, A., Callison-Burch, C., and Osborne, M. (2010). Stream-based Translation Models for Statistical Machine Translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '10, pages 394–402, Los Angeles, CA, USA. Association for Computational Linguistics.

[Levenberg et al., 2011] Levenberg, A., Osborne, M., and Matthews, D. (2011). Multiple-stream Language Models for Statistical Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 177–186, Edinburgh, Scotland, UK. Association for Computational Linguistics.

[Levy and Manning, 2003] Levy, R. and Manning, C. (2003). Is it harder to parse Chinese, or the Chinese Treebank? In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, ACL '03, pages 439–446, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Li et al., 2009] Li, Z., Callison-Burch, C., Dyer, C., Khudanpur, S., Schwartz, L., Thornton, W., Weese, J., and Zaidan, O. (2009). Joshua: An Open Source Toolkit for Parsing-Based Machine Translation. In *Proceedings of*

the Fourth Workshop on Statistical Machine Translation, WMT '09, pages 135–139, Athens, Greece. Association for Computational Linguistics.

[Littlestone, 1988] Littlestone, N. (1988). Learning Quickly When Irrelevant Attributes Abound: A New Linear-Threshold Algorithm. *Machine Learning*, 2(4):285–318.

[Locke and Booth, 1955] Locke, W. N. and Booth, A. D., editors (1955). *Machine Translation of Languages: Fourteen Essays*. MIT Press, Cambridge, MA, USA.

[Och, 2003] Och, F. J. (2003). Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, ACL '03, pages 160–167, Morristown, NJ, USA. Association for Computational Linguistics.

[Okita et al., 2012a] Okita, T., Rubino, R., and van Genabith, J. (2012a). Sentence-Level Quality Estimation for MT System Combination. In *Proceedings of the Second Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT-12)*, Mumbai, India.

[Okita et al., 2012b] Okita, T., Toral, A., and van Genabith, J. (2012b). Topic Modeling-based Domain Adaptation for System Combination. In *Proceedings of the Second Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT-12)*, Mumbai, India.

[Okita and van Genabith, 2011] Okita, T. and van Genabith, J. (2011). DCU Confusion Network-based System Combination for ML4HMT. In *Proceedings of the International Workshop on Using Linguistic Information for Hybrid Machine Translation (LIHMT 2011) and of the Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT)*, Barcelona, Spain.

[Papineni et al., 2002] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, ACL '02, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg,

V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Pierce et al., 1966] Pierce, J. R., Carroll, J. B., Hamp, E. P., Hays, D. G., Hockett, C. F., Oettinger, A. G., and Perlis, A. (1966). Languages and Machines — Computers in Translation and Linguistics. Technical report, Automatic Language Processing Advisory Committee (ALPAC), National Academy of Sciences, Washington, DC, USA.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

[Schwenk et al., 2006] Schwenk, H., Dechelotte, D., and Gauvain, J.-L. (2006). Continuous Space Language Models for Statistical Machine Translation. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, COLING '06, pages 723–730, Sydney, Australia. Association for Computational Linguistics.

[Scott, 1955] Scott, W. A. (1955). Reliability of Content Analysis: The Case of Nominal Scale Coding. *The Public Opinion Quarterly*, 19(3):321–325.

[Snover et al., 2006] Snover, M. G., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas*, AMTA '06, pages 223–231, Cambridge, MA, USA. Association for Machine Translation in the Americas.

[Snover et al., 2009] Snover, M. G., Madnani, N., Dorr, B., and Schwartz, R. (2009). TER-Plus: paraphrase, semantic, and alignment enhancements to Translation Edit Rate. *Machine Translation*, 23:117–127.

[Steinberger et al., 2006] Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., and Tufiş, D. (2006). The JRC-Acquis: A Multilingual Aligned Parallel Corpus with 20+ Languages. In *In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006*, LREC '06, pages 2142–2147, Genoa, Italy. European Language Resources Association (ELRA).

[Stolcke, 2002] Stolcke, A. (2002). SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, INTERSPEECH '02, pages 901–904, Denver, Colorado, USA. International Speech Communication Association.

[Tipping, 2001] Tipping, M. E. (2001). Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, 1(3):211–244.

[Torvalds and Diamond, 2001] Torvalds, L. and Diamond, D. (2001). *Just for Fun: The Story of an Accidental Revolutionary*. HarperBusiness, New York, NY, USA, 4th edition.

[van Genabith et al., 2012] van Genabith, J., Badia, T., Federmann, C., Melero, M., Costa-jussà, M. R., and Okita, T., editors (2012). *Proceedings of the Second Workshop on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid MT*. ML4HMT '12. The COLING 2012 Organizing Committee, Mumbai, India.

[Vapnik, 1995] Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer, New York, NY, USA, 2nd edition.

[Vilar et al., 2010] Vilar, D., Stein, D., Huck, M., and Ney, H. (2010). Jane: Open Source Hierarchical Translation, Extended with Reordering and Lexicon Models. In *ACL 2010 Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR*, WMT '10, pages 262–270, Uppsala, Sweden. Association for Computational Linguistics.

[Vilar et al., 2006] Vilar, D., Xu, J., D'Haro, L. F., and Ney, H. (2006). Error Analysis of Machine Translation Output. In *International Conference on Language Resources and Evaluation*, pages 697–702, Genoa, Italy.

[von Humboldt, 1836] von Humboldt, W. (1836). *Über die Verschiedenheit des menschlichen Sprachbaues und ihren Einflußauf die geistige Entwickelung des Menschengeschlechts*. Preußische Akademie der Wissenschaften, Berlin, Germany.

[Wittgenstein, 1922] Wittgenstein, L. (1922). *Tractatus Logico-Philosophicus*. Kegan Paul, Trench, Trubner & Co., London, UK.

[Wolf et al., 2011] Wolf, P., Bernardi, U., Federmann, C., and Hunsicker, S. (2011). From Statistical Term Extraction to Hybrid Machine Translation. In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation. Annual Conference of the European Association for Machine Translation (EAMT-11), May 30-31, Leuven, Belgium*, EAMT '11, pages 225–232, Leuven, Belgium. European Association for Machine Translation.

[Wu et al., 2012] Wu, X., Okita, T., van Genabith, J., and Liu, Q. (2012). System Combination with Extra Alignment Information. In *Proceedings of the Second Shared Task on Applying Machine Learning Techniques to Optimise the Division of Labour in Hybrid Machine Translation (ML4HMT-12)*, Mumbai, India.

[Zaslavskiy et al., 2009] Zaslavskiy, M., Dymetman, M., and Cancedda, N. (2009). Phrase-Based Statistical Machine Translation as a Traveling Salesman Problem. 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 333–341, Suntec, Singapore. Association for Computational Linguistics.

Vita

Christian Federmann was born in Cologne, Germany. He attended high school in Saarbrücken, Germany, and enrolled at Saarland University in fall semester 2001, pursuing a Diploma in Computer Science with a minor in Computational Linguistics. After completion of his Diploma in July 2007, he joined German Research Center for Artificial Intelligence (DFKI) as a full researcher. In 2009, Federmann started his doctoral studies while in parallel continuing his research work in several European research projects on Machine Translation and Knowledge Management. He contributed to the EuroMatrix and EuroMatrixPlus projects and led the development of META-SHARE, a shared repository of language resources, data and tools, developed as part of the T4ME Network of Excellence. In May 2013, Federmann joined the Computational Linguistics research group at Saarland University. After submission of his PhD thesis, he joined Microsoft Research in October 2013 where he works as part of the Microsoft Translator team.