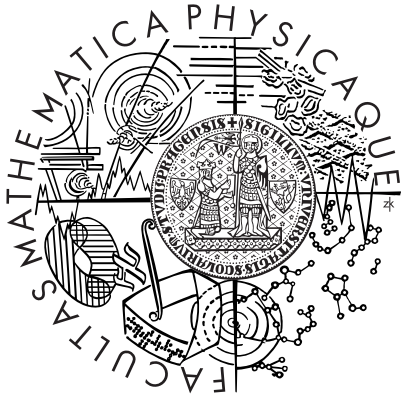
Charles University in Prague
Faculty of Mathematics and Physics

MASTER'S THESIS



Michal Novák

Machine Learning Approach to Anaphora Resolution

Institute of Formal and Applied Linguistics

Supervisor: Ing. Zdeněk Žabokrtský, Ph.D.

Study Program: Computer Science, Mathematical Linguistics

2010

I dedicate this thesis to my family, especially to my mother, who supports me and encourages me throughout my whole life.

I would like to thank my supervisor, Ing. Zdeněk Žabokrtský, Ph.D., for his patience and for his valuable expert advices. Moreover, I really appreciate that my supervisor and one of my friends, Mgr. Pavol Rusnák, provided me with a technical support for development and testing. This work would not be possible without the project of extended annotation of PDT led by Mgr. Anja Nědolužko and RNDr. Jiří Mírovský, Ph.D. I would like to thank them, too.

I declare I wrote my thesis by myself and listed all used references. I agree with making the thesis publicly available.

Prague, 5.8.2010 Michal Novák

Contents

List of Figures

List of Tables

Název práce: Rozpoznávání anafory metodou strojového učení

Autor: Michal Novák

Katedra (ústav): Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Ing. Zdeněk Žabokrtský, Ph.D.

E-mail vedoucího: zabokrtsky@ufal.mff.cuni.cz

Abstrakt: Rozpoznávání anafory je klíčové pro některé z úloh zpracování přirozeného jazyka (NLP), jako extrakce informací nebo dialogové systémy. Tato informace může byt hodnotná taky při strojovém překladu. Všechny předešlé práce týkající se rozpoznávání anafory v českém jazyce se soustředily především na zájmennou koreferenci. Díky nedávnemu projektu anotace širších anaforických vztahů v Pražském závislostním korpusu 2.0 však tato práce jde nad rámec zájmenné koreference. Pokouší se o rozpoznání koreference jmenných frází se specifickou referencí, generických jmenných frází a rozpoznání asociační anafory. Jsou v ní realizovány některé z nejúspěšnějších postupů v oblasti rozlišování anafor na základě strojového učení, konkrétně "ranking" a společné řešení úloh identifikace anaforu a nalezení antecedenta. Bylo vytvořeno množství rysů a analyzován jejích podíl na míře úspěšnosti. Nejlepší model koreference jmenných frází dosáhl F-hodnoty 39.4%.

Klíčová slova: rozpoznávání anafory, koreference jmenných frází, asociační anafora, Pražský závislostní korpus.

Title: Machine Learning Approach to Anaphora Resolution

Author: Michal Novák

Department: Institute of Formal and Applied Linguistics

Supervisor: Ing. Zdeněk Žabokrtský, Ph.D.

Supervisor's e-mail address: zabokrtsky@ufal.mff.cuni.cz

Abstract: Anaphora resolution is the key task for some of the Natural Language Processing (NLP) tasks like the information extraction or dialog systems. It can be also valuable in machine translation. All the previous works concerning the anaphora resolution in Czech language mostly focused on the pronoun coreference. Thanks to the recent project of the annotation of extended anaphoric relations in Prague Dependency Treebank 2.0 this work goes further. It attempts to resolve noun phrase coreference, identity-of-sense anaphora and part-whole bridging relations. It has adopted some of the state-of-the-art approaches in the area of machine learning approaches to anaphora resolution, particularly the ranking and the joint anaphor identification with the antecedent selection. It introduced a plenty of features and analyzed their contribution on the success rate. The best model of noun phrase coreference achieves the F-score of 39.4%.

Keywords: anaphora resolution, noun phrase coreference, bridging relations, Prague Dependency Treebank.

Chapter 1

Introduction

Anaphora resolution is an important area of research in Natural Language Processing (NLP). It plays a substantial role in more complex tasks as information extraction, question answering and machine translation.

The development of research in NLP is still often dependent on the data that are available, although there are experiments with unsupervised learning to change it. For some tasks the data can be obtained relatively easily, but there are also many of tasks that require the data manually annotated to the deeper layers of linguistic description. In anaphora resolution the latter is the case.

It is not so long that in currently the only data source for Czech language that incorporates the annotation of the underlying linguistic layers — Prague Dependency Treebank 2.0 anaphora was annotated mainly for pronouns. These days the second stage of anaphoric relations annotation extending them to nouns and to more complicated relations has almost finished.

To my knowledge, this work is the first one that explores the task of automatic resolution of these extended relations on the Czech data. Even though the available annotation is wider, this work concentrates mainly on noun phrase coreference and marginally on part-whole bridging relations.

In this chapter I proceed with introducing the fundamentals of theoretical background. Chapter 2 recalls some of the researches that have been conducted so far and emphasizes several different principles that have been used in the task of anaphora resolution. Whereas in Chapter 3 I describe the data I work with, Chapter 4 presents all the stages these data have to pass through to obtain the model and finally the results. Chapter 5 introduces the features derived from the data that serve as the basis for the model or that has been tested, but do not enrich the final model. In Chapter 6 I set down the evaluation metrics which is utilized to quantify the quality of models and I also stipulate the lower and upper bounds of this task. After that, everything is ready for experiments, analysis of the models and creation of final models and their evaluation in Chapter 7.

1.1 Basic terminology

Discourse is a unit of written text or speech, which consists of sentences. Two conditions on discourse have to be fulfilled — it must be coherent and cohesive. *Coherency* is the semantic unity of the discourse, its integrity concerning the deep structures and meaning of the discourse. *Cohesion* is the realization of coherency on the surface layer via the lexical and grammatical means [Nguy, 2006]. Let me illustrate these terms on the following examples:[1]

(1.1) The meeting between the Czech Minister of Foreign Affairs Karel Schwarzenberg with his German counterpart Guido Westerwellem in Berlin today had, officially, just a „tradition and courtesy“ character. The German minister of course confirmed their interest in the Lobkowicz Palace and #PersPron said that: „the site of our embassy in Prague is a historically valuable place for Germany“.

(1.2) Plus, why would the United States feel the need to take Hugo Chavez from power when his mandate is coming to an end?

Excerpts presented above individually satisfy the conditions of coherency and cohesion. Nonetheless, their concatenation is no longer a discourse. The concatenated text has no single consistent sense (coherency), what is confirmed by missing relations between the mentions that take part in the text — there is no mention related to United States or Hugo Chavez in the discourse 1.1.

Above the discourse, we can build a *discourse model*, the abstraction of reality. The model is made of *discourse entities* and their interactions. Discourse entity is an abstraction of real object and in a discourse it is realized by mentions (words or phrases). The relation between mention and its corresponding discourse entity is called *reference*. Reference can be divided into following types and subtypes:

- *exophora (deixis)* — reference to an object that has no other representation in a text
- *endophora* — reference within the discourse to another mention
  - *anaphora* — reference to a mention in the previous discourse
  - *cataphora* — reference to a mention in the following discourse

Since cataphora is far less frequent than anaphora,[2] in the following I concentrate only on anaphoric relations. The mention, which is pointing to the previous text, is called *anaphor* and the mention it refers to is an *antecedent*.

[1] The examples are taken from the articles in electronic version of Czech Focus newspaper (http://www.czechfocus.cz), namely “Jiří Lobkowicz: I am against selling the Czech heritage!” from 20.7.2010 and “Much Ado About Nothing...” from 26.7.2010.

[2] Empirical proof of this claim is depicted in Figure 4.1.

1.2 Categories of anaphora

Anaphora can be classified in various manners. Although one of the possible categorization is presented by Mitkov [2002], the individual categories overlap and some of them do not fit with the implementation I use. I present the classification similar[3] to that proposed in Nědolužko [2009], which fits the annotation of data I use, the Prague Dependency Treebank,[4] better and partially reflects the complexity of individual categories' resolution task. I distinguish between following categories of anaphora:

- coreference (identity-of-reference anaphora)
  - grammatical
  - textual
    - pronominal
    - noun phrase
- identity-of-sense anaphora
- bridging (associative, indirect) anaphora

If two mentions refer to the same entity, they are said to be in relation of *coreference*. The realizations of the same discourse entity are thus all mutually coreferential and form a *coreferential chain*. From this follows, that, said in terminology of discrete mathematics, the relation of coreference is an equivalency. Coreference is also denoted as *identity-of-reference anaphora*. On the other hand, the relation, when the anaphor and the antecedent do not target to the same object, but to the two different objects with similar description Mitkov [2002], is called *identity-of-sense anaphora*.

To illustrate the terms defined above, I return to the discourse from example 1.1. A discourse model of this discourse is briefly outlined in Figure 1.1. It includes discourse entities like [Karel Schwarzenberg], [Guido Westerwellem], [20.7.2010], [Lobkowicz Palace], [Germany] etc. The mentions "his German counterpart",[5] "Guido Westerwellem" and "The German minister" form a coreferential chain referring to the [Guido Westerwellem] entity. Relation between the mention "today" and the [20.7.2010] entity is a typical example of exophora, because the day when the article, this excerpt originates from, was published cannot be mined from the article (discourse) itself. Identity-of-sense anaphora can be seen between the pronoun "their" and the adjective "German" in the phrase "The German minister". Whereas the feature "German"

[3] Classifications differ only in the category of identity-of-sense anaphora. In Nědolužko [2009] it is treated as subtype of textual coreference.

[4] More on Prague Dependency Treebank and annotation of anaphora there in Chapter 3.

[5] This is an interesting observation, that to correctly identify this mention, one has to know the entity the word "his" refers to.

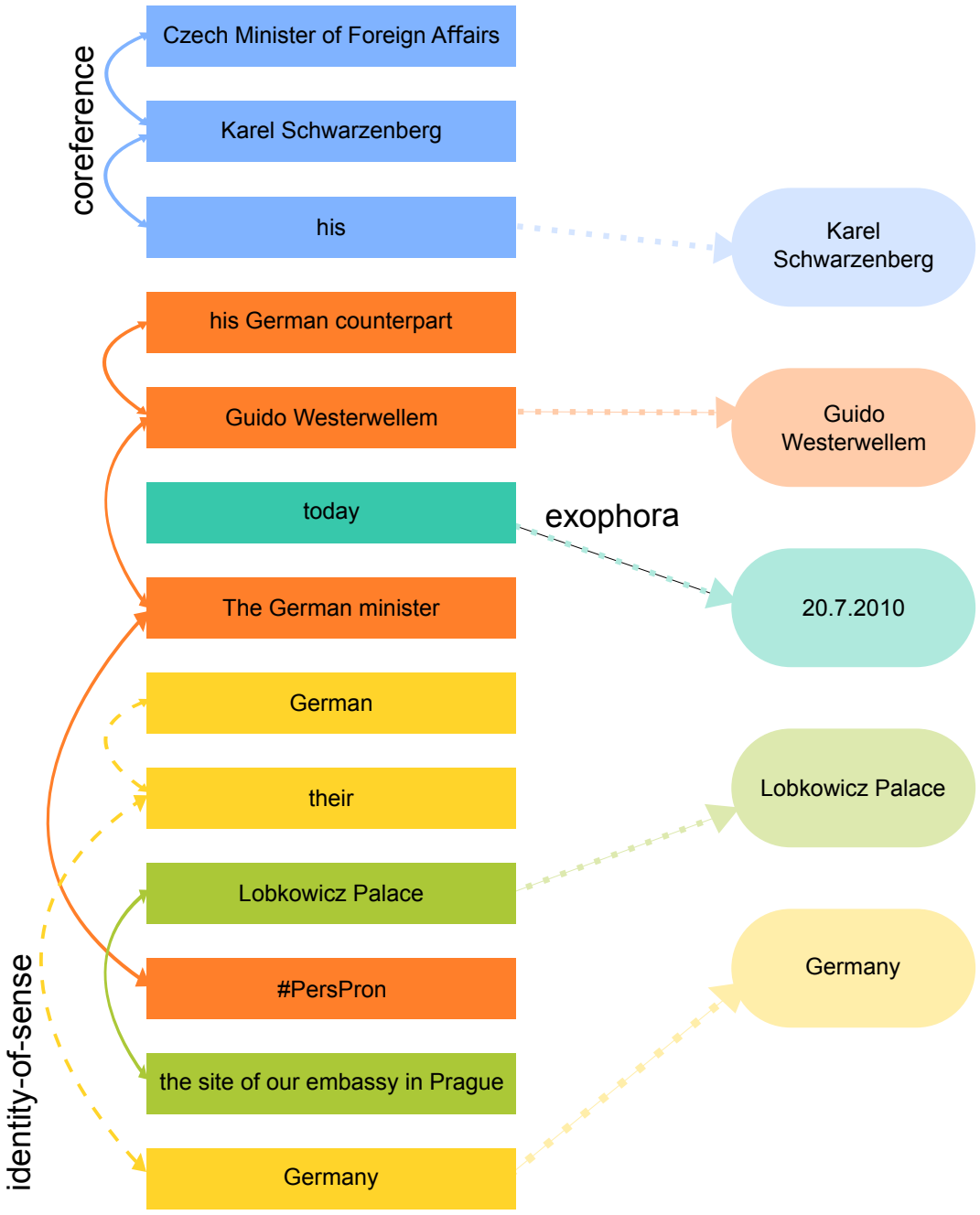


Figure 1.1: A discourse model of the excerpt 1.1. Not all discourse entities that figure there are depicted.

means that he is the minister of all German citizens, "their" can refer to another set of Germans (e.g. only the government).[6] However, these entities shares the notion of something representing Germany.

In Czech linguistics,[7] coreference is classified into *grammatical* and *textual coreference*. Grammatical coreference usually appears within a single sentence and its antecedent could be resolved by the grammatical rules of the language [Kučová and Hajičová, 2004]. On the other hand the textual coreference requires a context to resolve it. Textual coreference can be further classified into that with pronoun[8] anaphor (*pronominal coreference*) and that with noun phrase anaphor (*noun phrase coreference*). This distinction is important just from the point of complexity of their resolution.

In discourse individual entities relate together not only through the identity, but they are also connected via indirect types of relations. These relations are called *bridging (indirect, associative) anaphora*. Bridging relations are semantic or pragmatic relations that participate on coherency of the discourse [Nědolužko, 2009]. To illustrate, association anaphora covers the part — whole, subset — set, function — object relations and also relations between relatives ("mother" — "son"). Examples of this anaphora are in the section 3.2.

[6] Somebody else might not feel this distinction, what results in higher inter-annotator disagreement during the process annotation of these relations (see the section 6.2).

[7] Particularly in the theory of Functional Generative Description (see the section 3.1).

[8] It might be non-expressed.

Chapter 2

Related work

Anaphora resolution is one of the fundamental areas of research in Natural Language Processing, which has been studied for ages. An overview of history, how this task has developed, is nicely presented in Mitkov [2002].

In the last fifteen years the machine learning approaches are predominantly used to solve the task of anaphora resolution. Machine learning approaches can be classified into supervised and unsupervised.[1] Since my approach uses annotated data,[2] in next lines I describe previous works mainly from the area of supervised learning.

But I cannot conceal the unsupervised approaches, which appear still more and more frequently. To resolve pronoun anaphora, the work of Charniak and Elsner [2009] presents a generative model based on the sentence distance, person, number and gender probabilistic distributions along with the distribution of relation between the word and its head.[3] Another unsupervised approach proposed in Ng [2008] do not restrict itself only to pronoun anaphora, however the results are not comparable with resolvers which employ supervised models. Nevertheless, due to the more easily retrievable data without anaphoricity annotation, these approaches use, unsupervised learning is much more promising than present results show.

In the next sections I report some of the works concerning anaphora resolution to illustrate various techniques that have been acquired until now. Finally, I recall the systems achieving the best performance.

[1] And semi-supervised as something between.

[2] Except the feature approximating meronymy relation, which was extracted from the Czech National Corpus annotated only with morphological information (more in section 5.3.4).

[3] In Anglo-Saxon phrase structure theories, the head is the primary word in a phrase. The relation between "head of the phrase" and "the word belonging to the phrase" is equivalent to a dependency relation "parent" — "child" in European dependency grammars.

2.1 Classifier vs. Ranker

For a long time the problem of anaphora resolution was coerced into the category of classification problems. In *classification task*, one has to assign a class (tags, labels etc.) from the pre-defined set of classes to the currently processed object (word, sentence etc.). In case of anaphora resolution, this set of classes cannot be made of the possible antecedents because their number varies and many of them could not be connected with any of the candidates, because of their non-anaphoricity.

In order to retain handling this task as a classification task, another scheme was proposed. It was transformed into binary classification problem, where for the anaphora and one of its candidate, resolver has to assign one of the classes — "in relation" or "not in relation" [Denis and Baldridge, 2007a]. If the relation is a coreference, anaphor has to be connected with a unique antecedent,[4] whereas the binary classification allows more than one candidate to be assigned to the anaphor.

Selection of the antecedent is what distinguishes two works that follow the classification approach: Soon et al. [2001] and Ng and Cardie [2002]. While the former picks the closest candidate marked with the class "in relation" as the antecedent, the latter chooses that one, whose probability of being an antecedent is highest.

Although the latter approach compares the antecedent candidates, their concurrency is not considered in the stage of model training. This disadvantage is partially solved in the work of Yang et al. [2003]. It introduces a *twin candidate model*, where the instance consists not only of anaphor and antecedent candidate, but it contains also the second candidate to compete with the first one. For each instance in the training set, exactly one of those candidates has to be coreferential with the anaphor, so the training instances can be partitioned into those, where the first candidate is true antecedent (positive instances) and those, where the second candidate is the true antecedent (negative instances). During resolving, all candidates belonging to the particular anaphor are combined with the anaphor into triples and the resolver do the same partitioning on testing instances. Each time the instance is marked as positive, the first candidate gains a point, vice versa with negative instances. The candidate which is given the maximal score is singled out as the antecedent. Twin candidate approach is still a classification method, but incorporates pair concurrence into a trained model.

From twin candidate approach it is just a short step to a ranker. It is a fully competitive approach, all candidates of the anaphor with just some of them marked as in relation are trained together as a complex instance.[5] According to Denis and Baldridge [2007a], it completely abandons the classification view,

[4] On the other hand, bridging anaphora allows 1:N relation.

[5] In my work I denote it as *bundle.*

in which the feature combines the contextual predicate[6] with the class label, thus each feature has two versions, "in relation" and "not in relation" version. In case of rankers, features contain just the contextual predicate, no feature is associated with any class label, thus the features receive the weights on the basis of "how well they predict the correct output rather than correct label" [Denis and Baldridge, 2007a]. Besides the work [Denis and Baldridge, 2007a], followed by [Denis and Baldridge, 2008], where they incorporated maximum entropy ranker, this approach is also presented in Rahman and Ng [2009] (using SVM ranker) and using the Czech data in Nguy et al. [2009], the pronominal coreference resolver based on the perceptron ranker. All three researches confirm better results with ranker than using a classifier, Nguy et al. [2009] proposes the state-of-the-art resolver on Czech data, indeed.

2.2 Anaphor identification

Anaphora resolution can be treated as a sequence of two separate tasks: anaphor identification (determination) and antecedent selection. Whereas the former is responsible for selecting the mentions that are considered to be anaphors, the latter creates the link between the anaphor and the antecedent.

If the anaphora resolution is handled as classification task, one might not solve the problem of anaphor identification. The mention is labeled as anaphoric, when at least one antecedent candidate is declared to be coreferential with anaphor, otherwise it is marked as non-anaphoric. No explicit anaphoricity determination is acquired in resolvers of Ng and Cardie [2002] and Soon et al. [2001].

Although classifiers do not require a determination of anaphor as a specific step, it turned out that its incorporating results in performance improvements. An overview of methods used for this task can be found in Rahman and Ng [2009].

Nevertheless, one can see that anaphoricity resolution is tightly connected with antecedent selection. If there is no appropriate antecedent candidate, it is probable that the mention is non-anaphoric. In addition, even the certain antecedent cannot reject the wrong decision about the anaphoricity. Thus another approaches, which construct both models separately and finally jointly infer their decisions, were introduced. In Denis and Baldridge [2007b] the decisions are inferred using integer linear programming whereas the work of Ng [2009] transforms the task of the final inference to the problem of the minimum cut in graph.

In the data for ranking approach, there must be always at least one candidate that serves as the antecedent. Hence, a standalone anaphor determination has to be incorporated to filter out non-anaphoric mentions. Another approach was

[6] For example whether anaphor and antecedents shares the same lemma or if the antecedent's functor is ACT.

proposed in Rahman and Ng [2009], where the anaphor determiner and coreference resolver are trained at once into a single model. The bundle containing all antecedent candidates for anaphor is enriched with a special instance, that merely describe the anaphor candidate. If the ranker decides in favor of the special anaphor instance, it means that the mention is non-anaphoric.

2.3 Different approaches for different types of anaphora

In the section 1.2 I have shown the diverse taxonomy of anaphora. Some of the types differ substantially, so that treating them at once could harm the results. It is obvious that coreference has to be resolved using another model than bridging relations require.

Concerning coreference, many works (a short review is in [Denis and Baldridge, 2008]) handle this task with a single monolithic model regardless the anaphor type[7] and other easily mined features that could distribute this problem to sub-tasks. Work Denis and Baldridge [2008] shows that differentiation between tasks according to morphological type of anaphor could increase the performance of resolver. They train a special model for third person pronouns, a united model for first and second person pronouns, a model for definite descriptions,[8] a proper noun model and a united model for other types of anaphor. Connected with ranking approach or not, in both cases it produces significant improvement over monolithic approaches.

Nguy [2006] conducted a research on Czech data regarding pronominal textual and grammatical coreferences. She designed various approaches and trained separate models for different types of grammatical coreference, especially.

In Stoyanov et al. [2009] specialized models are not utilized. Nevertheless, they examine, how successfully their classifier resolves different types of anaphor. In differentiation they go further and divide both proper names and common noun phrases into those which exactly equals its antecedent's lemma, those which equals it only partially and those with no string match with its antecedent. Moreover, they introduce a special measure which on the basis of performances on these classes predicts the performance on another data set. It is calculated as weighted sum of observed performances of classes weighted by their proportion in the untested data set. They believe this measure can help in comparing the resolvers tested on different data sets.

Bridging relations have been so far less studied than coreferential ones, not only because it is much harder to resolve them, but also because it is often hard to define them reliably.[9] I mention here only one work of Poesio et al.

[7] For example its part-of-speech.

[8] Noun phrases often prefixed with "the".

[9] This fact is reflected in low inter-annotator agreement for bridging, for example in [Nědolužko et al., 2009].

[2004], which proposes the bridging references resolver. They employ lexical features consisting of WordNet distance between mentions and naturalness of the phrases of form "the word1 of word2" measured by Google queries, which approximate the meronymy relation. Furthermore, they apply salience features including the sentence distance and whether the antecedent is realized as a first mention.

2.4 State-of-the-art review

In this section I recall the results the selected previous works achieved in the area of coreference resolution.[10] However it is difficult to compare the success rates of their approaches because of various data sets they use, different degrees of golden standard information involvement and also because of differences in classes they attempt to resolve.

Concerning English data, the MUC-6[1995], MUC-7[1998] and ACE data sets in various versions (version ACE 2007 defined in [NIST, 2007]) are extensively used. These corpora contain the identity coreference annotations, other anaphoric relations have not been included. Comparison complications appears mainly with ACE data set, which is available in several versions and the authors, who conduct experiments on the corpus, partition the data into training and testing parts arbitrarily. For instance, Rahman and Ng [2009] carried out a selection of 599 documents from ACE 2005 whereas Denis and Baldridge [2008] uses a complete ACE-2 data set. The work Ng [2009] also uses the complete ACE-2 data set, but separates it into three independent models resulting in three different scores. In Haghighi and Klein [2009] they are provided with ACE 2004 subsets identical to those used in another two articles.

In addition, the task of coreference resolution do not work with a single evaluation measure.[11] Classical F-measure, MUC [Vilain et al., 1995], B^3 [Bagga and Baldwin, 1998] and ϕ_3-CEAF [Luo, 2005] scores are used instead. Fortunately, most of the works evaluate their models according to at least two of last three mentioned scores.

Another distinction in approaches which substantially influences the result is, whether *true mentions* or *system mentions* are used for training and testing. Mention is a linguistic item, which might hold an anaphoric relation. In case of corpora above, it stands for a noun phrase. These noun phrases are either manually (true mentions) or automatically annotated by a mention extractor (system mentions). Since in my work I focus just on anaphora resolution and I see the mention extraction task as a part of the deep-syntactic analysis, I use the gold standard syntactic analysis I am provided with in Prague Dependency Treebank. Therefore the results of related works I present in Table 2.1 are

[10] These works bother neither with identity-of-sense anaphora nor bridging relations.
[11] More about evaluation measures in the section 6.1.

Work	ACE			MUC-6		
	MUC	B^3	CEAF	MUC	B^3	CEAF
Ng and Cardie [2002] classifier, most probable antecedent	-	-	-	69.1	-	-
Yang et al. [2003] twin classifier	-	-	-	71.3	-	-
Denis and Baldridge [2008] ranker, separate models	71.6	72.7	67.0	-	-	-
Ng [2008] unsupervised	55.7 – 62.8 (51.6 – 57.8)		60.9 – 61.2 (55.7 – 59.6)			
Ng [2009] graph-cut inference	(59.4 – 63.9)	-	(59.4 – 63.8)	-	-	-
Haghighi and Klein [2009] syntactic and semantic constraints	(79.6)	(79.0)	(73.3)	(81.9)	(75.0)	(72.0)
Rahman and Ng, 2009 joint ranker	76.0 (69.3)	64.0 (61.4)	63.3 (59.5)	-	-	-

Table 2.1: F-scores of overall (noun phrases together with proper names and pronouns) coreference resolution on English data in previous works taking into account use of different datasets, metrics and degrees of gold standard usage. Values in parentheses stand for system mentions, otherwise they are true mentions.

those obtained on the true mentions and the results on the system mentions are printed in parentheses.

Individual works also differ in the types of anaphora they attempt to resolve. Some of them took into account the differences between anaphor types and unfortunately do not present the overall results. However, most of them present also the results without distinguishing coreferences with the pronoun, common noun or proper noun anaphor, or just did not conduct the separate tests.

In Table 2.1 I made an effort to take into account all the differences mentioned above and report the results presented in some of the related articles.

To my knowledge, until recently there were no Czech language data containing the noun phrase coreference or bridging relations annotation. The only source of the data with anaphora information was the Prague Dependency Treebank 2.0 (PDT), where the grammatical and pronominal coreference was annotated. Therefore all the previous works for Czech data focused on the grammatical or pronominal coreference.

In work of Němčík [2006], several classic rule-based algorithms were implemented and tested on PDT. The best approach achieved the F-score of 43.54%. Nguy [2006] utilized a machine learning approach, particularly the C4.5 top-bottom decision trees, to resolve various types of the grammatical and pronominal textual coreference separately. This approach reached the F-score of 75.8% for coreference with personal pronoun anaphor and 64.1% for coreference with possessive pronoun anaphor, respectively. In [Nguy and Žabokrtský, 2007] a rule-based approach was introduced again. Nonetheless, it outperformed both previous works with the overall F-score of 74.2% in resolving the coreference of personal pronouns, possessive pronouns and surface-deleted pronouns.

However, the state-of-the-art pronoun coreference resolver on PDT was presented by Nguy et al. [2009]. They implemented a perceptron ranker, which outperforms all previous approaches with F-score of 79.43%.

Chapter 3

Data description

In this chapter I describe the data I have been provided with to employ them in my anaphora resolver. The data has been extracted from the enriched version of Prague Dependency Treebank 2.0.

3.1 Prague Dependency Treebank 2.0

The *Prague Dependency Treebank 2.0 (PDT)* [Hajič et al., 2006] is a project for manual annotation of Czech data with linguistic information ranging from morphology to semantics and pragmatics. It is motivated by rich linguistic tradition in Prague and particularly by the theory of *Functional Generative Description (FGD)*. FGD is a language formalism based on a dependency syntax that represent the sentence as a system of mutually linked layers. This stratificational approach is also realized in PDT on a substantial collection of newspaper articles via following layers of annotation:

- *morphological layer* (m-layer) — the sentence is represented as a list of words (tokens); for each word it contains an information about lemma (base form), part of speech and grammatical categories (gender, number, case etc.)
- *analytical layer* (a-layer) — describes a surface syntax; tokens from the morphological layer are reflected here as nodes, which form a tree by connecting the nodes with dependency relations; several syntactic functions for these relations are introduced, for example predicate, subject, object and feature
- *tectogrammatical layer* (t-layer) — describes a deep syntax and partially semantics; the sentence is again represented as a dependency tree, but as opposed to the analytical layer, only auto-semantic words are reflected in the t-layer; on the other hand some nodes that have no representation on a surface level can be generated here, especially non-expressed members of the valency frame, e.g. a non-expressed personal pronoun; t-layer also

contains semantic functors of relation between nodes and topic-focus articulation information

The t-layer is also the place, where coreferential relations are annotated. In its original version, PDT merely contains the annotation of grammatical coreference and textual coreference with a pronoun as an anaphor.

3.2 Extended anaphora relations in PDT

For this work, which by the means of supervised learning tries to create an automatic resolver mainly of textual coreference with a noun as an anaphor, the original annotation of PDT was insufficient. Thus I have to use the extension of PDT, the results of the next stage of annotation that is still being conducted by the working group led by Mgr. Anja Nědolužko and RNDr. Jiří Mírovský, Ph.D [Nědolužko et al., 2009]. The extension adds an annotation of other relations that secure coherence of the text, including non-pronominal anaphor coreference, exophora and bridging relations.

Extended anaphora in PDT can link a wider variety of elements (mentions) than in MUC and ACE[1] — full noun phrases, anaphoric adverbs ("Prague" — "there"), numerals ("2010" — "this year"), clauses and sentences if they co-refer with a noun phrase ("I asked him whether . . . " — "my question") and adjective just in case they are coreferential with a named entity or a nominal head ("German" — "Germany") Nědolužko et al. [2009].

Textual coreference (feature `coref_text`) is marked between mentions with identical referent, denoted as coreference with specific reference [Nědolužko et al., 2009] and labeled with value `type0`. In the beginning of annotation, the annotators distinguished the coreference whose arguments were synonymous phrases (value `SYN`) and also hyperonymous phrases (value `ER`). Nědolužko [2009] claims that around 10% of PDT was annotated in this way. Such detailed annotation was soon abandoned because of its time complexity and drop in inter-annotator agreement. Data I worked with were also partially annotated in this detailed way. Hence, to make my data consistent I had to unify coreferences of type `SYN` and type `ER` and include them into the category of `type0`.

Identity-of-sense anaphora is also annotated in extension of PDT. It is annotated as a part of textual coreference feature with value `NR` and denoted as coreference with generic reference [Nědolužko et al., 2009]. In spite of the fact that in the section 1.2 I did not include this type of anaphoric relation into the category of coreference, I comply the notation in PDT and denote it as `NR` coreference.

Two other types of references are annotated by a feature `coref_special`. First of them is an endophoric reference to a discourse segment of more than

[1] The corpora used for English (see the section 2.4).

one sentence and the second is an exophoric link. None of them has the antecedent marked.[2]

In case of bridging relations, following types are annotated:[3]

- part of the whole — values `PART_WHOLE` and `WHOLE_PART`
- subset/element of the set — values `SUBSET_SET` and `SET_SUBSET`
- function on the object — values `P_FUNCT` and `FUNCT_P`

  (3.1) government — prime minister

- coherence relevant discourse opposites — value `CONTRAST`; for example:

  (3.2) *People*$_1$ don't chew, it's *cows*$_1$.

- non-co-specifying explicit anaphoric relation — value `ANAF`; for example:

  (3.3) "*Rainbow*$_1$?" The priest put the finger *on this word*$_1$, so that he didn't forget, where he stopped.

- other relation — value `REST`; this contains relations between relatives ("mother" — "son"), event — argument relations ("listening" — "listener") etc.

An example of annotated sentence depicted in a PDT tool TrEd is presented in Figure 3.1. Further information about annotation of extended anaphora in PDT can be found in Nědolužko et al. [2009] and in Nědolužko [2009], respectively.

3.3 Necessary modifications of the PDT data

For the purpose of machine learning, data should be divided into three groups:

- training data — used to train the model,
- development testing data — used as testing data during development and improving the model,
- evaluation testing data — used as testing data for the final model evaluation.

[2] In case of discourse reference it is just for the time being.
[3] The examples are borrowed from Nědolužko et al. [2009].

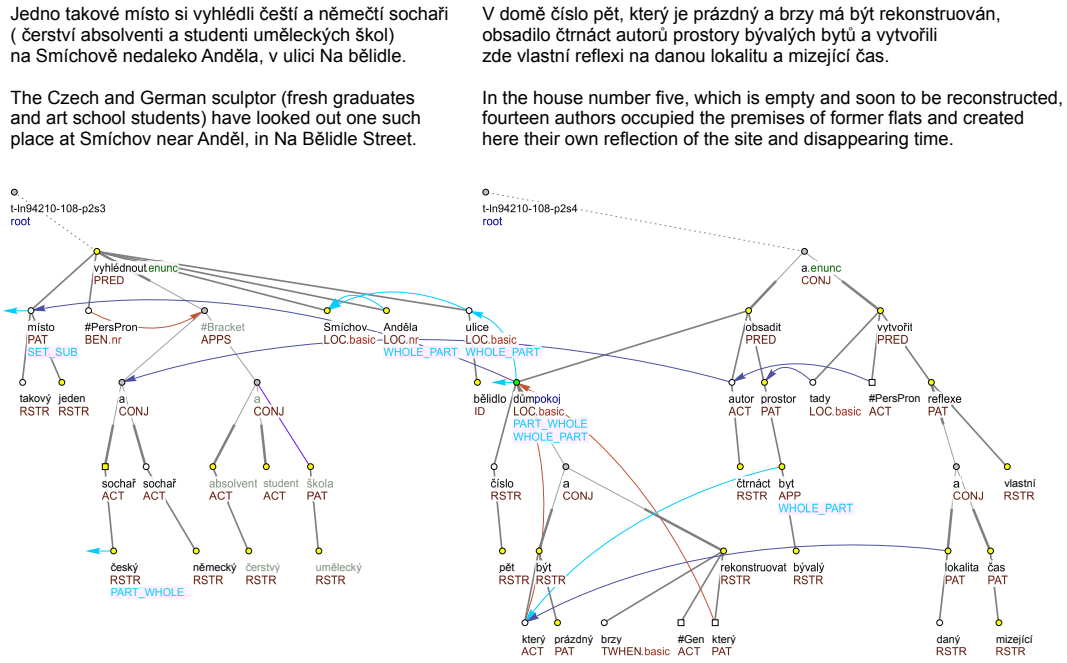


Figure 3.1: An example of the sentences annotated in the extended PDT, mutually connected by anaphoric relations containing the grammatical coreference (red arrows), textual coreference (blue arrows) and bridging relations (cyan arrows).

	PDT partitions
training data	`train-1`, `train-2`, `train-3`, `train-4`
development data	`train-5`
evaluation data	`train-8`

Table 3.1: My own partitioning of extended PDT data I have been provided with. Individual partitions can be found on the CD attached (see Appendix B).

Although the original data in PDT are divided in this manner, in the time, I carried out experiments, the annotation of extended discourse relation had still not been accomplished. Thus, I had to perform my own division. Which PDT partitions form which group of data set according to this division is described in Table 3.1.

Whereas the PDT data are in PML format,[4] the ranker used for experiments requires data to be a list of bundles, where the bundle is a list of instances stored in a tab-separated table format. Therefore the data extracted from the PDT has to be transformed in a preprocessing stage described in Chapter 4.

[4]Prague Markup Language (PML) [Pajas and Štěpánek, 2006] is XML-based markup language designated for storing rich linguistic annotations; it was designed during the annotation of PDT.

	training data				development data				evaluation data	
	complete		reduced		complete		reduced			
all	97988		16384		25751		4096		21396	
type0	13774	14.0%	2694	16.4%	3527	13.7%	603	14.7%	2876	13.4%
NR	3520	3.6%	409	2.5%	1187	4.6%	107	2.6%	933	4.3%
PART_WHOLE	422	0.4%	59	0.3%	96	0.3%	14	0.3%	91	0.4%
WHOLE_PART	1091	1.1%	120	0.7%	229	0.8%	31	0.7%	168	0.7%

Table 3.2: Number of bundles in the data tables used during the experiments.

Due to performance complexity when training a model from the whole training data, I decided to reduce the number of bundles used for training during development approximately to one sixth. The development testing data were reduced respectively. These data subsets were chosen from the beginning of individual groups of data and their sizes are tabulated in Table 3.2. In the following text, I denote these data as *reduced (training, development) data.*

Chapter 4

Resolver implementation

In this chapter I describe the whole process that data from PDT has to pass through to either create a model of them or make an attempt of resolving the anaphora in them. Whole process can be divided into the following stages:

1. Data preprocessing
   (a) Basic instances extraction
   (b) Basic instances filtering and pairing
   (c) Features adding
   (d) Features filtering
2. Model learning
3. Anaphora resolving

The purpose of the preprocessing stage is to extract the relevant information from the PDT data and transform it into a format that can be consumed by the resolver. In order to facilitate and accelerate experiments, I divided the preprocessing into four separate sub-stages. Each of the consecutive sub-stage reads the data resulting from the previous one, modifies them in some way and outputs the intermediate data, which are an input for the next sub-stage. The first sub-stage reads the files of PDT data whereas the output of the last sub-stage is ready for resolver to create a model from it.

The whole preprocessing stage is implemented in Perl, which is extensively used in applications concerning natural language processing. Modules, which are used in scripts are build on Moose[1] framework that offers easy object oriented programming in Perl. Model learning and resolving stage are provided by an external application (more in the section 4.5).

[1] http://www.iinteractive.com/moose/

4.1 Basic instances extraction

Starting with documents of PDT, the first sub-stage of preprocessing performs their transformation from the PML format into a tabular format. For each document in a data set, every single node in the tectogrammatical representation of the document is processed. From the t-layer the corresponding nodes on the a-layer and the m-layer can be accessed, so for a particular tectogrammatical node this sub-stage is able to print out all features, which are later used for training or as a source for transformation into another set of features. Thus each line of the output table represents one t-layer node and I denote this line with features as a *basic instance*. This extraction sub-stage is implemented as a query for the PDT tool btred[2], which provides a traversal through documents and nodes. The query script can be found in `queries/all_nodes.btred`.

4.2 Basic instances filtering and pairing

The second sub-stage is implemented in `create_data_table.pl` and it is the most complicated among all sub-stages in the chain they form.

The following sub-stage ensures a creation of *instances*. Instances used for machine learning in the task of anaphora resolution are often formed as a combination of two basic instances, one describing the word, which can be later resolved as anaphor (*anaphor candidate*) and the other that can be its antecedent (*antecedent candidate*).[3] However, the words cannot be combined arbitrarily, this approach would result in a quadratic explosion of the data. Furthermore in most cases the resulting pair would be pointless, for example, when the words come from different documents. Frequently the pairing of two words that do come from the same document, but which are too far apart would be useless. This fact is confirmed by a distribution of sentence distances between the anaphor and the antecedent candidate shown in the graph in Figure 4.1. The graph also shows the rare occurrence of cataphora. Thus the anaphor candidate is combined only with the words which precede it and belong to current or immediately preceding sentences. Maximum number of preceding sentences, i.e. size of the sentence window, can be modified via an input parameter. I decided to ignore cataphora, which helped to implement this second sub-stage in the way to run faster. The constraint of the sentence window was also used in the development and evaluation data with the same size of the window as in the training data, hence all experiments in the following sections are conducted on the data constrained in this way.

[2] http://ufal.mff.cuni.cz/ pajas/tred/

[3] In the following text I often abbreviate these terms just to *anaphor* and *antecedent*, meaning the candidates for these roles. Then for the real or predicted anaphor I say that "anaphor (candidate) is (predicted as) anaphoric" or denote the mention as "true (predicted) antecedent", respectively.

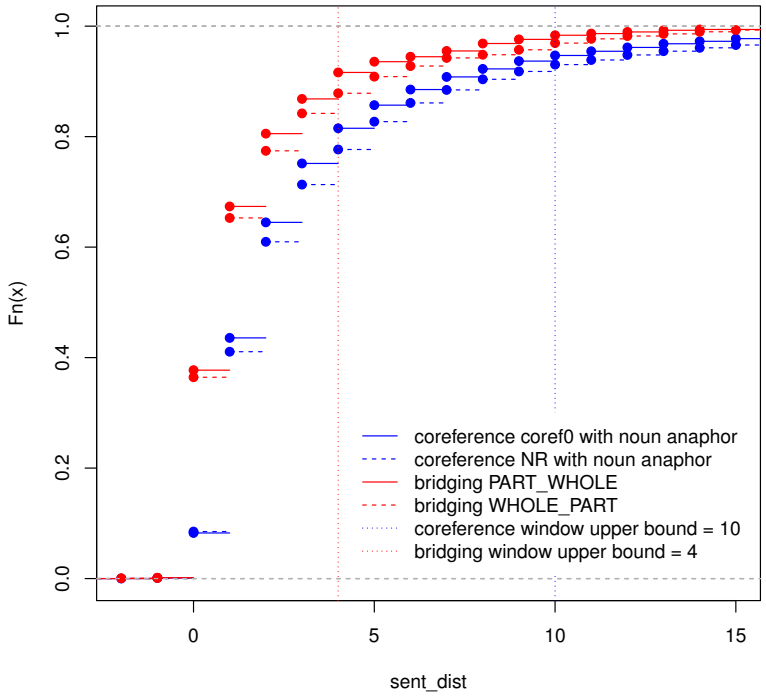


Figure 4.1: Empirical cumulative distribution function of the sentence distance between the anaphor and the antecedent in the training data, measured on all types of anaphora this work concentrates on.

This sub-stage is also responsible for anaphor and antecedent filtering. In the section 2.3 I showed on related works, that in some cases it is better to distinguish between different types of anaphor. In this work focusing mainly on the noun phrase coreference it is necessary to restrict the part of speech of them to being only nouns. In Figure 4.2 showing a distribution of antecedents' part of speech one can see that it might not have a negative effect to restrict the part of speech of antecedent candidates, too. Both filters can be set by parameters and the system is ready to work with any other different filters, if necessary.

As I already mentioned, each anaphor candidate is combined with preceding words to form the group of instances. This group is identified by the anaphor candidate and we call it a *bundle*. Each instance that belongs to the bundle carries the information whether the two words which form the instance are in relation we want to automatically resolve (coreference, bridging etc.), or not. This key information[4] is extracted just during this sub-stage into the feature `is_rel`. If there is no word in the sentence window that is in relation with anaphor candidate, the anaphor candidate is considered non-anaphoric.[5] This fact is then reflected in a special instance describing only the anaphor candidate, we call it an *unary instance*. The unary instance is inserted into each bundle as the first instance and for non-anaphoric anaphor candidates its

[4]This information is key particularly in testing, when this has to be compared with the predicted one. In supervised learning it is crucial also for training.

[5]In fact, its antecedent may lie further than the size of window or it may be cataphoric.

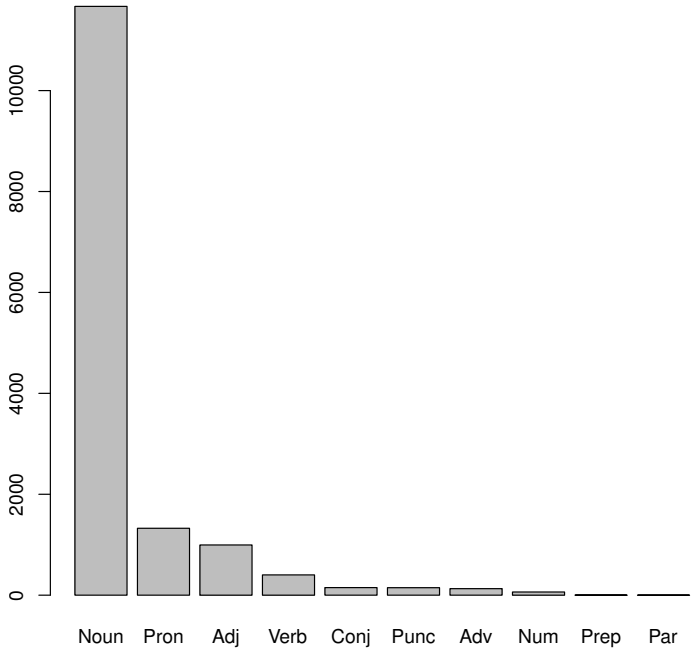


Figure 4.2: Part-of-speech of the antecedent distribution in `coref0` coreference with a noun phrase anaphor measured on the training data.

key feature is set to false value, otherwise to true one. In other words, the data are prepared for joint resolution of both anaphoricity and anaphora link as it is proposed in Stoyanov et al. [2009].

In the process of combining, the choice of whether the word is an anaphor or not is influenced by the antecedent filtering. If the true direct antecedent of the word was something that was filtered out, this word would be mistakenly labeled as non-anaphoric. This can be avoided if the antecedent is part of a coreferential chain. We can follow the coreferences until we find a node, which passes the antecedent filter. If no such node exists, then the anaphor has to be marked as non-anaphoric. It looks like I am employing the information which I want to resolve, but it is not the case. I am just exploiting the fact that coreference is symmetrical and transitive relation, so it does not matter which antecedent I found. In case of filtering the antecedent to be only nouns, it helps to prevent the fails, when the direct antecedent is a generated node but it points to another noun through a grammatical coreference link or the direct antecedent is a pronoun and the noun antecedent lies in the end of the coreferential chain. Trying to resolve noun to pronoun coreference is much harder task than resolving the pronoun to noun coreference together with the noun to noun coreference followed by deriving the noun to pronoun coreference link just from the rule of symmetry.

Although an addition of features does not belong to primary functions of the second sub-stage, several ones are added. These are the features which need either the sentence window, for instance to retrieve a surface distance between

anaphor candidate and antecedent candidate, or a document context, for example to find out how often the word currently to be processed appears in the document.

The last thing, this sub-stage is responsible for, is limiting the number of bundles in the output, which is important to deal with the performance issues during the development (see the section 3.3).

4.3 Adding features

The third sub-stage of the preprocessing (in `add_attribs.pl`) reads the bundles and the instances they contain and enriches the table with some new features. The new features are all derived from the feature set served by the previous sub-stage or from already added new features. In addition, for derivation of some features the external models are used. For example, a model of synonyms, a lemma of the anaphor and a lemma of the antecedent are obligatory elements to create a new feature describing whether the anaphor and the antecedent are synonymous.

Moreover, features are not only added but also modified. It mainly concerns the features whose values are continuous. Their values are quantized, where the quantization clusters are defined on the basis of distribution of the training data. Quantization is carried out in order to simplify the model and hopefully also to improve the success rate of resolving.

4.4 Feature filtering

The previous sub-stage results in creating a table with a variety of features, but some of them useless for training, like technical features represented by document or node ID. It also included the features with a lot of rarely occurring values associated, for example both anaphor and antecedent lemma. Some of them has to be removed just because their presence decreases a success rate of the model.[6] Script `filter_columns.pl` does this feature filtering, so its output is ready to serve as a source data for making a model.

4.5 Learning and resolving

In this moment all the required preprocessing of data tables has been finished and the system is ready to create a model from them. Variety of algorithms can be used for machine learning, for example decision trees, support vector

[6] This is probably caused by the imperfection of the algorithm which train the parameters of the model.

machines or perceptrons. In my approach I have chosen the *maximum entropy* modeling.

Maximum entropy learning has been extensively used in area of natural language processing. It is motivated by Occam's razor principle that the simplest solution is usually the correct one. In case of machine learning it means that it does not make any assumptions about things that were not observed in the data. In other words, while staying in the boundaries of selected properties of the data, model has to be as uncertain as it is possible.

The properties of the data are the features. Features are usually binary-valued functions that describe data. Thus it is easy to convert the multi-valued features which are output by the preprocessing stage into the binary ones required by the learning algorithm. The boundaries the model has to preserve are called *constraints*. They are defined by equation 4.1 of the expected value p of the feature f_j in the model and its expected value $\widetilde{p}$ in the training data [Berger et al., 1996].

$$p = \sum_x \Pr(x) f_j(x) = \sum_x \widetilde{\Pr}(x) f_j(x) = \widetilde{p} \qquad j = 1, \ldots, m \tag{4.1}$$

To be of greatest possible uncertainty is to reach the maximum entropy. From the theory of Lagrangian multipliers one can derive that the model which maximizes its entropy has the following form:

$$\Pr(x) = \frac{\exp(\lambda_1 f_1(x) + \cdots + \lambda_m f_m(x))}{\sum_x \exp(\lambda_1 f_1(x) + \cdots + \lambda_m f_m(x))} \tag{4.2}$$

Then the model learning consists of searching the λ_i parameters, which have to be set in the way that the model complies the constraints based on the training data.

In this work I utilize ranking modification of maximum entropy modeling presented by Denis and Baldridge [2007a], which fits the task of anaphora resolution better than its standard classification version.[7] Since I train the joint model for both anaphoricity determination and anaphor link resolution, I did not have to prepend a special anaphoricity identifier to the ranker. I used the implementation of ranker in Toolkit for Advanced Discriminative Modeling (TADM)[8] which computed the parameters by the Improved Iterative Scaling algorithm [Berger et al., 1996]. TADM[9] requires the data to consist of binary-valued features in special numeric, human unreadable format. Although it provides the Python interface, which do these necessary transformations and facilitates the work with this software, in original version it contained several

[7] For the comparison of ranking and classifying approach see the section 2.1.

[8] I also made an attempt to deploy the Perl maximum entropy module `AI::MaxEntropy`, which failed because it was impossible to force it to work as a ranker. Furthermore, instead of TADM and its prerequisites which are mostly implemented in compiled languages C++ and Fortran, this interpreted Perl module is substantially slower. Thus I decided for TADM, despite its more complicated installation.

[9] http://tadm.sourceforge.net/

	anaphor filter	ante. filter	window size (sentences)
coreference	nouns	nouns	current + 10 previous
bridging	nouns	nouns	current + 4 previous

Table 4.1: Setting of parameters during data preprocessing used for all experiments.

bugs. Therefore in deployment of my anaphora resolution system on the CD attached to this work the patched version of this software is included as well and it cannot be replaced by the version available on the website.

This interface also allows to easily launch the resolution and to print out the feature weights as well as predicted probabilities for all candidates in a bundle. Since in this work I mainly concentrate on the coreference resolution I allowed the resolver to pick only one instance in a bundle, the one with the highest score. However, this simplification can possibly harm the bridging resolution, because the anaphor can be connected with several different antecedents via different relations.

4.6 Resolver parameters for experiments

In the sequential data processing and modeling I have just introduced, there are three parameters that has to be assigned: anaphor filter, antecedent filter and the sentence window size. All these three parameters put a limit on the model. The purpose of such limiting are to improve the time and memory complexity of the whole process, to possibly improve the model by restricting it to be used only for some cases (anaphor and antecedent filtering) as well as to reduce the data sparseness on minimum while not harming the results too much (sentence window size). In this section I present the setting of these parameters I used during all experiments.

Since the main task of this work is to conduct a research in the area of noun phrase coreference, the choice for the anaphor filtering is clear — it has to be a noun phrase, e.g. the head of the phrase must be a noun. According to Figure 4.2 showing the part-of-speech distribution of antecedents, the antecedents with a noun head are far most frequent. Thus I do the same filtering also on antecedents. For the bridging relations I adopted the same approach and filter both the anaphor's and the antecedent's head to be a noun exclusively. This simplification is done mainly because of time complexity. Working with the unfiltered data would spend much more time during the experiments.

From Figure 4.1 we can see that over 90% of antecedents is covered in the current and several previous sentences. Bridging relations tend to be more local than coreference ones, so I decided to set the size of the window to the

current sentence with 4 previous sentences for all types of bridging anaphora and the current one with 10 previous ones for all types of coreference.

The parameters setting is summarized in Table 4.1.

Chapter 5

Construction of features

In this chapter I introduce and describe the features which I hope may help in the task of anaphora resolution. Whether they really helped and how much can be found in section 7.1. Features are extracted from the various layers of PDT and for some of them external information is used.

Features can be categorized according to many aspects. As I mentioned in the section 4.2, each instance is a combination of two basic instances, one pertaining to anaphor candidate and another one to antecedent candidate. Furthermore, mostly in the third sub-stage of preprocessing features related to both words are included. Thus, I distinguish two types of features:

- unary features - related only to either anaphor or antecedent candidate,
- binary features - related to both anaphor and antecedent candidate.

However, the categorization I adopted as the main one, assigns categories according to semantic differences between the features. In the following text I describe these categories and the features belonging to them.

All the features to be introduced only relate to the head nodes of anaphor or antecedent mentions. Thus the terms anaphor and antecedent are used mainly for their heads since this moment.

5.1 Distance features

All candidates to antecedent are chosen from the strictly defined window of sentences preceding the anaphor candidate (see the sections 4.2 and 4.6). This is the basic restriction on the distance. Moreover the information about position within this window could have a serious effect on resolver performance. So that some features expressing the various types of distances between anaphor candidate and antecedent candidate were introduced.

The first included distance feature is simply the sentence distance (`sent_dist`), which describes how many sentence boundaries lie between the

anaphor and the antecedent. This number is non-negative, where zero means that they lie in the same sentence.

Nevertheless, the sentences can vary in their length. Thus, a distance should be expressed by a measure with softer granularity. A word distance seems to be a better approximation. In this work I am dealing with two types of word distance. They differ in the type of word ordering used for the computation of the distance.

Surface word distance (`word_dist`) is computed using the surface word ordering present on the morphological layer of PDT. It simply represents how many words appear between the anaphor and antecedent. Punctuation marks are counted as individual words.

Deep word distance had to be calculated from the tectogrammatical layer of PDT, because deep ordering is one of the necessary element for its calculation. Deep ordering sorts the words by their communicative dynamism. The words, whose discourse entities are already known from the previous sentences are less dynamic than those unseen yet. This distance is represented by the `deep_word_dist` feature.

Both word distance features are quantized. The quantization mapping was manually defined on the basis of histograms describing a distribution of these features.

5.2 Grammatical features

Grammatical features included in the training tables can be divided into two groups, based on their origin, the layer they were extracted from.

From the morphological layer of the PDT three basic features were extracted — morphological tag (`anaph_m_tag`, `ante_m_tag`), lemma (`anaph_lemma`, `ante_lemma`) with its suffix (`anaph_lemma_suffix`, `ante_lemma_suffix`). Only the former one can be classified as grammatical feature, the latter two features are considered to be lexical and are described in the section 5.3.1.

Morphological tag is 16 characters long string, where each position represents one morphological category (part of speech, gender, number, case, person, tense, negation flag, aspect etc.). In our case number, gender and negation flag belonging to the anaphora candidate are extracted and either concatenated or compared with corresponding categories retrieved from the antecedent candidate. It results in three concatenation features (`both_numbers`, `both_genders`, `both_negs`) and three Boolean features carrying an information, whether the participating categories are equal or not (`numbers_equal`, `genders_equal`, `negs_equal`).

The t-layer of PDT provided training tables with some grammatemes (`anaph_t_gender`, `ante_t_gender`, `anaph_t_number`, `ante_t_number`) and functors (`anaph_functor`, `ante_functor`).

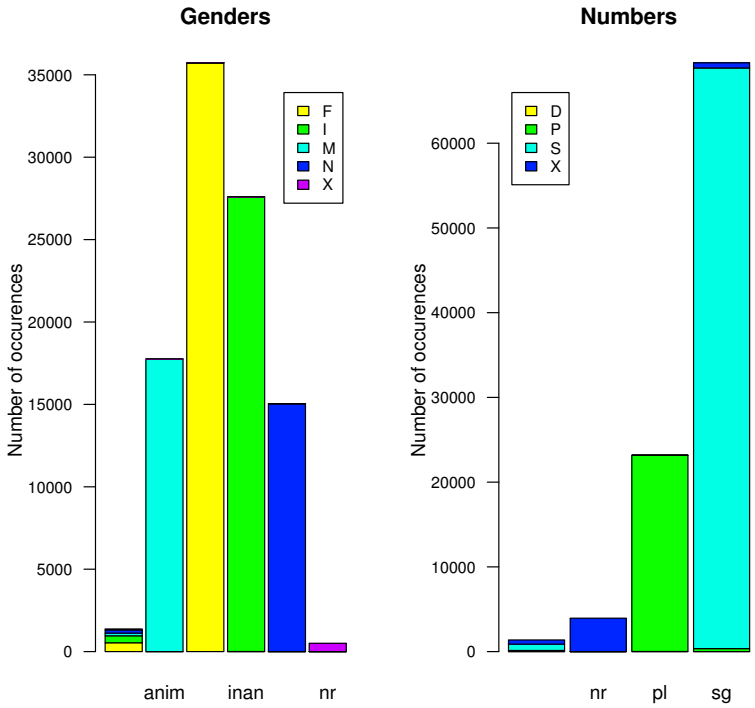


Figure 5.1: Correlation between tectogrammatical grammatemes and corresponding morphological categories in the training data.

Grammatemes are tectogrammatical equivalent of the morphological categories. As you can see in Figure 5.1, there is very strong positive correlation between categories on m-layer and t-layer. However, there are some differences, caused by fact that grammatical rules do not reflect the semantics, for example the quantity when concerning the number. Here is an example of one of the differences between numbers on these two layers:[1]

(5.1) jedny dveře [m/number=pl, t/number=sg] (=one door)
dvoje dveře [m/number=pl, t/number=pl] (=two doors)

While grammatemes strongly correlate with morphological categories in most of experiments they were left out and no other features were derived from them.

Functors represent the semantic values of syntactic dependency relations Mikulová et al. [2006]. For experiments I used the values of functors alone as well as functors of anaphor and antecedent candidates concatenated (`both_functors`).

5.3 Lexical and related features

Intuitively, one can feel that in the tasks, which I have attempted to construct an automatic resolver for, lexical features will play a more significant

[1] This example is borrowed from Mikulová et al. [2006].

Lemma (Czech)	Lemma (English)	Lemma suffix	Suffix description
maso	meat	_^(jídlo_apod.)	explanation — it is food
Bonn	Bonn	_;G	geographical name
Martinův-1	Martin's (meaning nr. 1)	_;Y_^(*4-1)	given name, comment on derivation — remove last 4 chars and add "-1" to get the original word ("Martin-1")

Table 5.1: Examples of lemmas and their lemma suffices in PDT.

role than grammatical features. Unlike the task of pronominal anaphora resolution, where gender and number agreement are the most important, because the pronoun or the non-expressed pronoun is just a syntactic placeholder for a noun, in noun phrase coreference and bridging resolution the features concerning lexical semantics are extremely useful.

5.3.1 Base features

All these lexical features are derived from the lemma (`anaph_lemma`, `ante_lemma`) and lexical information (`anaph_lemma_suffix`, `ante_lemma_suffix`) of the lexical item. On the morphological layer of PDT [Hana et al., 2005] this information is, if present, stuck to the lemma as suffix (see examples in Table 5.1) and during the first sub-stage of preprocessing it is separated into two features mentioned above.

Lemma represents a base form of the word, for nouns it is nominative singular form, for verbs it is infinitive. Lemma feature itself has too many rarely occurring values to serve as a reliable contributor to the model. In spite of that it can be employed to expand the feature set with more useful features.

5.3.2 Equality of lexical items

A short browse through the training data gives us a nice insight that there is almost 64% of cases, when the lemmas of two coreferential nouns are identical. This fact is simply reflected by a Boolean feature (`lemmas_equal`), which is true, if the lemmas are equal, otherwise it is false. Frequently the mention closest to the anaphor is the antecedent, therefore I added a feature (`lemmas_equal_dist_rank` which joins identity of lemmas with their surface distance (more on distance features in the section 5.1) in a way that the closest mention fulfilling the condition of lemmas' equality is assigned number one, the second closest number two etc. Exactly the same technique is used for creation of another distance ranking features described later, changing just the filtering condition.

5.3.3 Synonymy of lexical items

Two coreferential nouns might not be the same, they can be also synonymous. To find out, whether two words are synonymous, we need a dictionary of synonyms. There are many ways how to obtain such a dictionary. The most reliable but I think the most complicated solution is to integrate a manually created dictionary.[2] If some similar electronic dictionary exists, it often does not provide a machine-readable output and is not available for free. Another solution is presented in the article of Agirre et al. [2009], where they adopted a context window approach to search for the words with similar meaning. For each word they collect a window surrounding the word and group together the words that appear in the center of the same window. These groups then form the similarity classes. To achieve reliable results, this approach requires a huge mass of data. Although I had access to Czech National Corpus - SYN2005[CNC, 2005] which contains about 100 millions of words, it is just a drop in the bucket compared to the corpus of 1.6 trillions of words used in Agirre et al. [2009]. Therefore I have chosen another approach. I incorporated a dictionary retrieved from a translation model.[3] The Czech-English translation model was extracted from the Czech-English Parallel Corpus (CzEng) [Bojar et al., 2009] by the unsupervised method of word alignment proposed in Och et al. [1999]. It consists of translation pairs, from which the Czech words sharing the same English translation were retrieved and grouped together.

The dictionary of synonyms gets involved as the external model in the third sub-stage of preprocessing. It receives two words and returns just the Boolean value, whether these two words are synonymous or not. This information is stored in the feature `lemmas_synon`. Berger et al. [1996] demonstrates that the usage of disjunction of two features could sometimes improve the model more than including them each alone. Thus, despite the presence of both `lemmas_equal` and `lemmas_synon` features, I enriched the feature set with their disjunction (`lemmas_equal_synon`). Similarly as in case of feature `lemmas_equal`, a distance ranking feature is created (`lemmas_equal_synon_dist_rank`, where this disjunction acts as the filtering condition. A distance ranking based exclusively on synonymy has not been included.

5.3.4 Meronymy / holonymy of lexical items

Aside from synonymy, which helps to reveal the coreference relations, another semantic relations between lexical items might be crucial for the task of bridging resolution. I focused on the meronymy / holonymy, which is a part-whole / whole-part relation. For example, the ceiling is a meronym of the room and conversely the room is a holonym of the ceiling.

[2] For example the dictionary of synonyms of Pala and Všianský [2000].

[3] This module was implemented by Zdeněk Žabokrtský.

Correspondingly to synonymy, two approaches to gain a dictionary of part-whole relations are possible: utilize manually annotated data or extract it from unannotated data in an unsupervised way. The former approach is represented by the semantic networks lead by the most famous one — WordNet and its language mutations containing also Czech — EuroWordNet [Vossen, 1998]. Due to more complicated access and the well-known problems of WordNet described in Poesio et al. [2002] I avoided it and adopted the latter, more robust approach based on the approach introduced in the same article. They used syntactic patterns like possessives or noun phrases with the preposition "of" to acquire the meronymy information. I adjusted this method to fit the Czech language principles and employed the Czech National Corpus (CNC) — SYN2005 to obtain this information. I have been searching for all phrases starting with noun in any case followed by zero or one preposition "z/ze" ("of/from"), then followed by zero or more non-prepositions and non-nouns in genitive and ended with a non-plural noun in genitive. In the query language of CNC the pattern looks like following:

(5.2)
```
[tag="N.*"][lemma="z"]?[tag="[^RN]...2.*"]*
[tag="N..[^P]2.*"]
```

In many cases, phrases following this pattern materializes the part-whole relation between the nouns in the pattern, where the first (on the left side) acts as a part and the second (on the right side) acts as a whole. Example 5.3 shows the phrase that agrees with the proposed pattern. On the other hand Example 5.4 confirms the necessity of the constraint that the last word must not be in plural.

(5.3) strop$_{PART}$ pokoje$_{WHOLE}$ (the ceiling$_{PART}$ of the room$_{WHOLE}$)

(5.4) sbírka$_{SET}$ dokumentů$_{SUBSET}$ (the collection$_{SET}$ of documents$_{SUBSET}$)

Applying the query expressed by this pattern on SYN2005 yields a model, which consists of ordered pairs of nouns representing a part and a whole and of the additional information, how many times the particular words in defined roles co-occurred. While the model of synonymy returns only the yes/no answer, the model of meronymy provides the data tables with a probability-like measure. For two selected lemmas, the model can be asked for either measure of that the first lemma is a part and the second is a whole (P_{PW}) or measure of that the first lemma is a whole and the second is a part (P_{WP}). The relation between P_{PW} and P_{WP} is following:

$$P_{PW}(lemma1, lemma2) = P_{WP}(lemma2, lemma1) \tag{5.5}$$

Thus I will confine myself only to description of how whole-part measure, P_{PW}, is calculated. At first, two contributing probabilities, I denote them as p_1 and p_2, have to be calculated. Probability p_1 is the conditional probability that

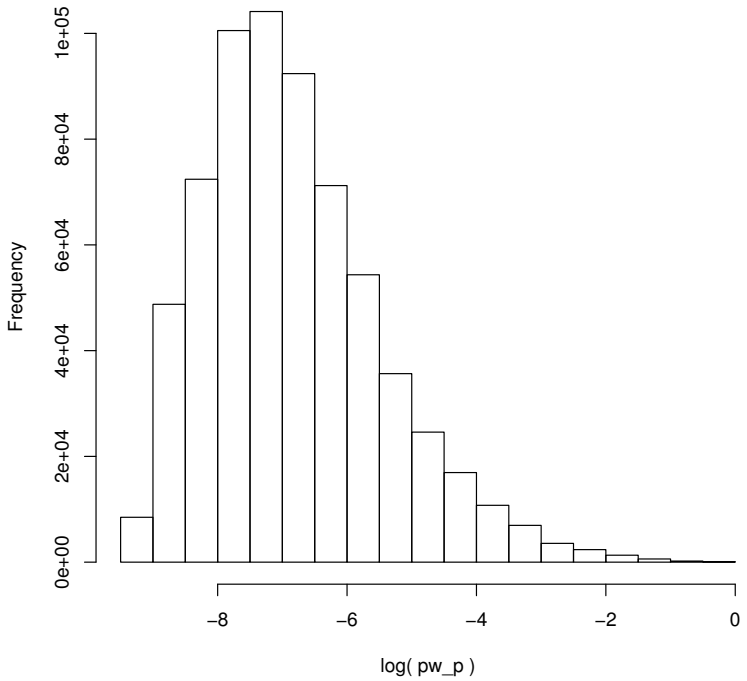


Figure 5.2: Distribution of non-quantized non-zero values of `pw_p` feature depicted on the logarithmic scale.

*lemma*1 is a part and *lemma*2 is a whole given *lemma*1 is a part (equation 5.6). Probability p_2 is the conditional probability that *lemma*1 is a part and *lemma*2 is a whole given *lemma*2 is a whole (equation 5.7). Then the P_{PW} is a harmonic mean of p_1 and p_2 as it is shown in equation 5.8 and that is why I do not denote it by a word "probability".

$$p_1 = \Pr(PART = lemma1 \wedge WHOLE = lemma2 | PART = lemma1) \quad (5.6)$$

$$p_2 = \Pr(PART = lemma1 \wedge WHOLE = lemma2 | WHOLE = lemma2) \quad (5.7)$$

$$P_{PW} = \frac{2p_1p_2}{p_1 + p_2} \quad (5.8)$$

However, the variables P_{PW} and P_{WP} are continuous, what is not ideal for the training method I use. Therefore the values have to be quantized into discrete values (categories). The most frequent value is zero, which accounts for 83.3% of the training data. Thus it is mapped on its own discrete value. The categories for other values were chosen manually on the basis of histogram estimating the distribution of non-zero P_{PW} values (Figure 5.2). Quantized values for P_{WP} are the same because of the equation 5.5. In conclusion, quantized P_{PW} and P_{WP} variables result in two new features (`pw_p` and `wp_p`).

5.3.5 Frequency of lexical items

The original data in the PDT are divided into documents. Coreferential or bridging link can appear only within one document, a link cannot target into the document different from that it originates from. Imagine a document, for example a newspaper article.[4] Some entities in the document, represented on the surface level as lexical items, are usually mentioned more frequently than others, they account for a significant part of document's topic. It is highly probable that this entity will be mentioned again. On the other hand, another ones occur occasionally and conversely, the probability that these entities will appear again is lower.

I tried to exploit this fact, thus I introduced two features, both defined only for antecedent candidate. The first one (`ante_count_in_doc`) expresses the absolute number of occurrences of the antecedent in the document. While the absolute numbers may differ across the documents, the rank based on this feature should be more stable. Thus, the second feature from this category (`ante_rank_in_doc`) is a rank, where for the most frequent word it is equal to one, for the second two etc. If there are more words with the same frequency, they are assigned the same rank and the word with next lower frequency is assigned this rank increased by one.

These features require the context of other words in a document, hence they are added already in the second sub-stage of preprocessing, the stage of combining the basic instances (see the section 4.2).

5.3.6 Named entities

Named entities have a special position in a discourse model. The words that usually refer to them are proper nouns. There is almost no ambiguity in assignment of a proper noun to its corresponding named entity in a discourse model. My observation is that wherever the proper noun appears, it shows to the same entity in most of the cases. This property can be exploited to improve the model by enforcing it to treat the proper nouns and common nouns in different ways. In the section 2.3 I presented the work of Denis and Baldridge [2008], where they showed that treating the anaphora with various anaphor types in separate models can improve the model. My work follows this approach partially because it only focuses on the relations with the noun phrase anaphor. However, for the time being I decided not to treat proper and common noun anaphors separately, though include various features concerning named entities.

In order to work with features which recognize the proper nouns or the forms attached to named entities, I have to introduce a couple of methods to retrieve this information. The simplest approximation of named entity recognition is

[4] In PDT the documents are newspaper articles, but it can be a chapter in a book, in other words some longer coherent excerpt.

Type	Explanation
Y	given name
S	surname
E	member of a particular nation or territory
G	geographical name
K	company, organization, institution
R	product

Table 5.2: List of the named entity types concerned in the features.

to declare each lexical item, whose lemma starts with a capital letter, to be a proper noun. I introduced a Boolean feature denoting whether anaphor's lemma starts with an upper-case (`anaph_cap`), a feature for antecedent's lemma respectively (`ante_cap`). Another Boolean feature `both_cap` is true only if both anaphora and antecedent start with capitals. As I postulated above, two proper nouns with equal lemmas are likely to refer to the same entity. Thus I included the `lemmas_equal_both_cap` feature, which is true only if both participating features `lemmas_equal` and `both_cap` are true. This feature also served as a condition for a ranking similar to the rankings introduced in the sections 5.3.2 and 5.3.3 (`lemmas_caps_dist_rank`). In the following sections I denote this group of features as *capital features*.

Moreover, I have been provided with a gold standard. In PDT each named entity has been manually classified subject to its type (the list of types is tabulated in Table 5.2) and this information has been stored in the lemma suffix [Hana et al., 2005]. I have used this manual annotation to derive the set of features corresponding to capital features (`anaph_ne`, `ante_ne`, `ne_equal`, `lemmas_equal_nes_equal`, `lemmas_nes_dist_rank`). In the following these features are denoted as *gold named entity (NE) features*.

If there is no gold standard available, some named entity recognizer could be used. Stoyanov et al. [2009] has shown that this change does not influence the result significantly.

Chapter 6

Metrics and boundaries

6.1 Evaluation metrics

During experiments I had to know how the trained model performs, in other words I wanted to find out how well the predicted key feature denotes that the contributing words are in a relation (coreference, bridging).

One of the measures extensively used in computational linguistics is *accuracy*. It is expressed as the ratio of correctly predicted instances to all predicted instances (equation 6.1).

$$A = \frac{\#correctly\_predicted}{\#all\_predicted} \tag{6.1}$$

However, often this measure is rather distorting. In our case, pairs which are not in relation occur much more frequently than those, which are.[1] But accuracy includes the couples, which are not in a relation and the prediction follows it, in the same category of correctly predicted as the correctly predicted pairs that are in a relation. Thus the correctly negative instances, which we are not interested in, account for a large proportion in a success rate around 90%, this evaluation approach leads in.

Since the key feature `is_rel` is binary I can incorporate evaluation measures which express a success rate in this task better. These measures are *precision* and *recall*. The former describes how many of predicted positive instances are correctly positive while the latter stands for how many of all positive we have correctly predicted (equations 6.2 and 6.3). Although sometimes it is important to handle these two measures separately, the success rate should be expressed by a single number. *F-measure* as a harmonic mean of precision and recall fulfills this requirement (equation 6.4), thus I use this measure throughout the experiments.

[1] See Table 3.2.

Real	Predicted	Category
0	0	-
0	$i, i \neq 0$	*predicted_positive*
$i, i \neq 0$	$j, j \neq i$	*all_positive*
$i, i \neq 0$	$j, j = i$	*correctly_predicted_positive* *predicted_positive* *all_positive*

Table 6.1: Mapping of the output from resolver to categories figuring in the precision and recall measures calculations.

$$P = \frac{\#correctly\_predicted\_positive}{\#predicted\_positive} \tag{6.2}$$

$$R = \frac{\#correctly\_predicted\_positive}{\#all\_positive} \tag{6.3}$$

$$F = \frac{2PR}{P+R} \tag{6.4}$$

As I have described in the section 4.2 the instances belonging to one particular anaphor candidate are grouped into a bundle. The bundle also includes an unary instance symbolizing that this candidate is non-anaphoric. A ranking algorithm then for each bundle chooses the instance which is the most likely among the others. For each bundle the resolver outputs an sequence number of the predicted instance. Hence, the calculation of the categories appearing in the formulas 6.2 and 6.3 had to be altered. For example, the category of correctly predicted positive bundles consists just of the bundles, whose instance marked to be in a relation is the same in prediction as well as in reality. The complete mapping is shown in Table 6.1.

Now I will focus just on coreferential relations. There are many approaches how to annotate this relation because coreference is an equivalence, so it follows the principles of reflexivity, symmetry and transitivity.[2] A set of nodes in coreference, which is in a discrete mathematics terminology a class of equivalence, can be annotated in PDT. It forms a coreferential chain, where each item refers to the previous one according to the tectogrammatical ordering. Another approach could be that each node would point merely to the first one due to some ordering. These different ways of annotation complicate the comparing of the results of researches based on the so annotated data. For this reason special evaluation measures for coreference and similar tasks were introduced, for example MUC [Vilain et al., 1995], B^3 [Bagga and Baldwin, 1998]

[2]However, the principle of transitivity is weaker. It is common that after sequence of consecutive relations, the meaning slightly changes.

or ϕ_3-CEAF [Luo, 2005] score. Despite this advantage I favor the less complicated evaluation measure sufficiently satisfying the needs of the coreference task, the F-measure.

The advanced measures introduced above cannot be used in their original versions for bridging resolution task nonetheless. The whole-part and set-sub relations are definitely not symmetric likewise the contrast relation is not transitive at all.

6.2 Lower and upper bounds

Before conducting the experiments using maximum entropy model constructed of many features I had to know, what is the baseline success rate, i.e. the success rate which can be achieved by a trivial method. This baseline should stand for a lower boundary, which should not be fallen behind.

For the task of coreference resolution I designed two baseline algorithms. The first one is based on the `lemmas_equal_dist_rank`. If there is at least one antecedent candidate with lemma equal to the anaphor's, the closest one according to surface distance is chosen, otherwise the anaphor is declared to be non-anaphoric. Since the participants of coreference can be also synonymous, the second approach is to choose the closest word that is synonymous with anaphor. The third approach is the combination of previous two, e.g. the closest word that is equal or synonymous is picked.

Considering bridging, it is much more difficult to choose a sensible baseline. Finally, I have again selected the feature that from the point of view of lexical semantics characterizes it the most. Due to the fact I have tested only the bridging types part-whole and whole-part, I based the baseline algorithms upon the `pw_p` and `wp_p` features. As the antecedent the closest mention to anaphor with non-zero value of part-whole measure P_{PW} [3] is picked (for whole-part respectively). If there is no such mention, anaphor is declared not to be in the relation.

Baseline algorithms have been tested on the reduced development data[4] for all types of relations this work is concerned in. The results are shown in Table 6.2.

The results of baseline coreference resolution show the best performance using the equality approach with precision being outperformed by recall. In case of the `NR` coreference the precision is almost three times lower than it is on the `coref0` coreference, what corresponds with the three times smaller number of `NR` coreference links in the development data (see Table 3.2).

[3] Defined in the section 5.3.4.

[4] The reduced development data were used to ensure that results of baseline approaches can be compared with another development experiments described in the section 7.1. Nevertheless, the best baseline approaches were eventually tested also on complete development and evaluation data and results are presented in the section 7.3.

	Precision	Recall	F-measure	Precision	Recall	F-measure
	coref0 coreference			NR coreference		
equality of lemmas	31.70	60.03	41.49	5.60	59.81	10.25
synonymy of lemmas	5.23	4.48	4.83	1.55	7.48	2.57
equality or synonymy of lemmas	25.50	61.53	36.05	4.74	64.49	8.83
	PART_WHOLE bridging			WHOLE_PART bridging		
part-whole approximation	0.11	21.43	0.22	0.00	0.00	0.00
whole-part approximation	0.15	28.57	0.30	0.11	9.68	0.23

Table 6.2: Results of baseline approaches on the development data for all types of anaphora this work is concerned in.

	coreference	bridging
1st measurement (40 sent.)	67%	42%
2nd measurement (40 sent.)	41%	52%
3rd measurement (100 sent.)	68%	57%
4th measurement (100 sent.)	65%	39%

Table 6.3: Inter-annotator agreement on textual coreference with a nominal anaphora and bridging measured on data from PDT by Nědolužko et al. [2009].

The baseline results of bridging relations are particularly interesting, although their success rates are nearly, in one case exactly, at zero. The approach tailored to the resolution of the WHOLE_PART bridging works better also on the PART_WHOLE bridging.[5] Even though the baseline approaches used for bridging simplify the pw_p and wp_p features just to a binary classifying, it is an unmistakable sign that these features do not play the role I have awaited.

On the other side of the scale than baseline is an upper bound. It should be almost impossible to statistically significantly surpass it. One way to specify the upper bound for textual coreference with a nominal anaphor is to think about it as the intuitively more difficult task[6] than resolving of coreference with a pronominal anaphor is. The best result for pronominal anaphora resolution in Czech achieved in Nguy et al. [2009] is F-score of 79.43%.

Another, more frequent, solution is to represent the upper bound by an *inter-annotator agreement* (IAA), which is utilized to measure a reliability of manual annotation. In Nědolužko et al. [2009] they conducted 4 measurements of IAA between 2 annotators on the subset of 280 sentences from the same data I have been provided with. Into Table 6.3 I extracted their results of the IAA (measured by F-measure) on the arguments of relation as well as on its type

[5] In fact, the tests on the complete development data shows that part-whole approximation baseline with 0.47% solves the WHOLE_PART bridging resolution better than the whole-part approximation with 0.11%, so it is completely reversed.

[6] The reason why the noun phrase coreference is harder to resolve than the pronoun coreference could be based on the assumption that the features originating from the lower layers of linguistic description do not contribute as substantially as it is in the case of pronouns. Worse resolvable lexical semantics instead could play a significant role there.

for both textual coreference and bridging.

Since the data, which served as a source for IAA measurement, equal to the data, I use, the results from the IAA accord with requested upper bound more than the intuitive bound described above. Therefore I pick the best of the presented IAA measurements as the upper bound, 68% for coreference and 57% for bridging.

Chapter 7

Experiments and evaluation

The features I have introduced in the previous section now should be combined to a *training feature set*, which serves as an input into the stage of model training using the maximum entropy ranker defined in the section 4.5. In this section I propose the models for resolving both coreference and bridging anaphora and publish the evaluation results of these models. In addition I take a look into the inners of models and empirically prove the contribution rate of selected features.

7.1 Development experiments and model analysis

Instead of presenting here all the intermediate results I will rather concentrate on selected features and I will try to show here, how various types of information influence the final performance and how some of the features correlate.

Most of the analyses have been conducted on the models different from the final model, because these analyses were carried out in the time of searching the best feature set and as an effect some of the intermediate models as well as the final model have been discovered during the analysis. If all the analysis tests were provided on the final model, the absolute numbers of the evaluation could vary but the trends would remain the same.

All the following tests have been carried out on reduced development data with models for `type0` coreference trained on the reduced training data.

7.1.1 Complete feature set model

The complete set (I denote the set and the model made of it as `full_set+base`) consists of following 43 features:

	Precision	Recall	F-measure
full_set	57.4	39.1	46.6
full_set+base	48.9	22.6	30.9

Table 7.1: Comparison of the model trained on the reduced development data from the all features and the same model extended with unbound domain features. The profound impact of these features is obvious.

anaph_lemma, anaph_lemma_suffix, anaph_sempos, anaph_t_gender, anaph_t_number, anaph_functor, anaph_m_tag, ante_lemma, ante_lemma_suffix, ante_sempos, ante_t_gender, ante_t_number, ante_functor, ante_m_tag, deep_word_dist, word_dist, sent_dist, ante_count_in_doc, ante_rank_in_doc, pw_p, wp_p, lemmas_equal, lemmas_equal_dist_rank, lemmas_synon, lemmas_equal_synon, lemmas_equal_synon_dist_rank, both_functors, both_numbers, both_genders, both_negs, numbers_equal, genders_equal, negs_equal, anaph_cap, ante_cap, both_cap, lemmas_equal_both_cap, lemmas_caps_dist_rank, anaph_ne, ante_ne, ne_equal, lemmas_equal_nes_equal, lemmas_nes_dist_rank

The model full_set+base also includes the base features whose positive contribution to the model I have doubted in the section 5.3.1 — anaph_lemma, ante_lemma, anaph_lemma_suffix and ante_lemma_suffix. These features have an enormous domain of values, but many of the values occur just once or several times and therefore they cannot have such an impact on the model as those that occur in hundreds or thousands. On the contrary, huge amount of features with marginal impact causes a deformation of the model. This fact can be nicely shown on the performance comparison between the model full_set+base and the model derived from the former by removal of base features (full_set), depicted in Table 7.1.

From Table 7.1 we can see that on the reduced development data the above model full_set achieves the F-score of 46.6%. It is clearly better result comparing to that of the baseline in Table 6.2. Another interesting observation is that whereas the recall of baseline outperforms its precision, in the complete model it is the opposite. Even though in the task of coreference resolution it cannot be said that the one F-measure participant is strongly more important than the another one like the precision is in the task of web search in particular, higher precision is slightly preferred to higher recall. From this point of view, the full_set model is also of higher quality comparing to baseline, even if they reached the same F-score.

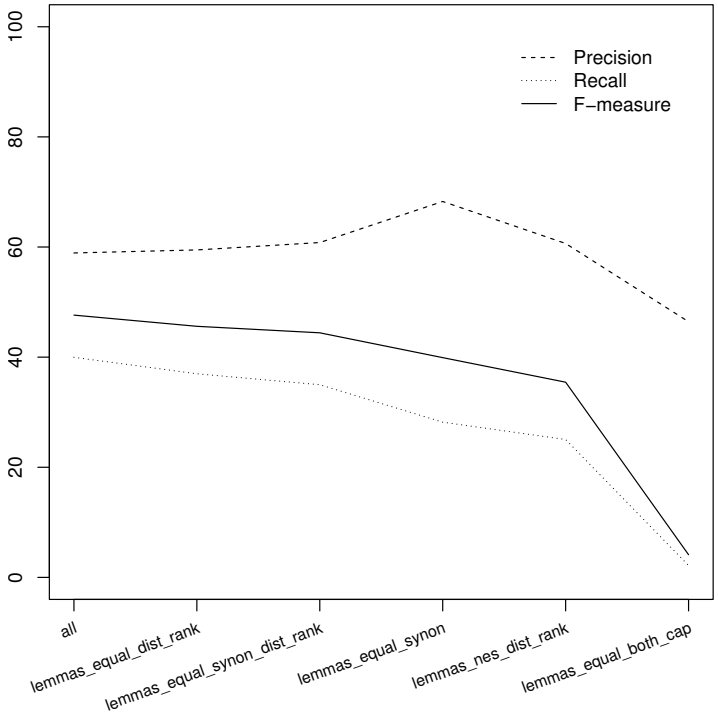


Figure 7.1: Performance changes when sequentially removing equality and synonymy features from the model `inter1_set`

7.1.2 Effect of lemmas' equality and synonymy

In the section 5.3 I stated that lexical properties of words is priceless information in area of resolving text coherence relations. Particularly equality or synonymy of words in the case of coreference with nominal anaphor can affect the model significantly.

To prove this, I have prepared the following set of experiments. I started with the following feature set `inter1_set`:

`ante_functor, ante_m_tag, deep_word_dist, word_dist, sent_dist, ante_rank_in_doc, pw_p, wp_p, lemmas_equal_dist_rank, lemmas_equal_synon, lemmas_equal_synon_dist_rank, both_functors, numbers_equal, genders_equal, negs_equal, ante_cap, lemmas_equal_both_cap, ante_ne, lemmas_nes_dist_rank`

Sequentially I removed one by one all the features related to equality or synonymy.[1] Figure 7.1 illustrates this process and shows the quality of the models that is indeed decreasing. There are three interesting observations I have to point out:

[1]The order of their removal proceeded from the most generic to the most specific while preferring the ranking features. When it was complicated to follow this rule, order of the features was chosen arbitrarily.

	Precision	Recall	F-measure
deep_word_dist + word_dist	58.92	39.97	47.63
deep_word_dist	58.44	39.64	47.23
word_dist	59.21	39.97	47.72
none	57.47	37.65	45.49

Table 7.2: Contribution of word distance features on inter1_set model.

The relevancy of equality and synonymy features accords with my assumption. While this information is present in the model, its quality decreases more or less uniformly. After removal the last feature related with equality — lemmas_equal_both_cap, the F-score falls by over 30%.

Features lemmas_equal_dist_rank, lemmas_equal_synon_dist_rank and lemmas_equal_synon share the big portion of information. In results and in Figure 7.1 it has expressed by larger drop after their elimination.

Increasing precision in the first half of experiments is the last interesting fact. Removal of features proceeded from those more general (e.g. lemmas_equal_dist_rank) to those more specific (e.g. lemmas_nes_dist_rank). The last two eliminated features especially are connected only to the proper nouns and named entities, hence the model trained before their removal boils down into resolving merely the coreference of named entities.

7.1.3 Effect of distance

Another kind of features that in the previous text I have declared to be much more valuable than the others are the distance features. They consist of one sentence distance feature and two word distance features measured on different layers of annotation.

Whereas the surface word distance can be easily retrieved, there is no automatic tool for resolving the deep ordering which tectogrammatical word distance depends on. In PDT the information about deep ordering was annotated manually. From this reason it is important to find out how the presence of one or the other of the distances influences the model. In Figure 7.2 you can see the correlation between non-quantized values of features word_dist and deep_word_dist. It shows almost the linear dependence between these two distances, which suggests that deep_word_dist could be a redundant feature.

To prove this I conducted the experiment, where the referential model inter1_set containing both word_dist and deep_word_dist features has

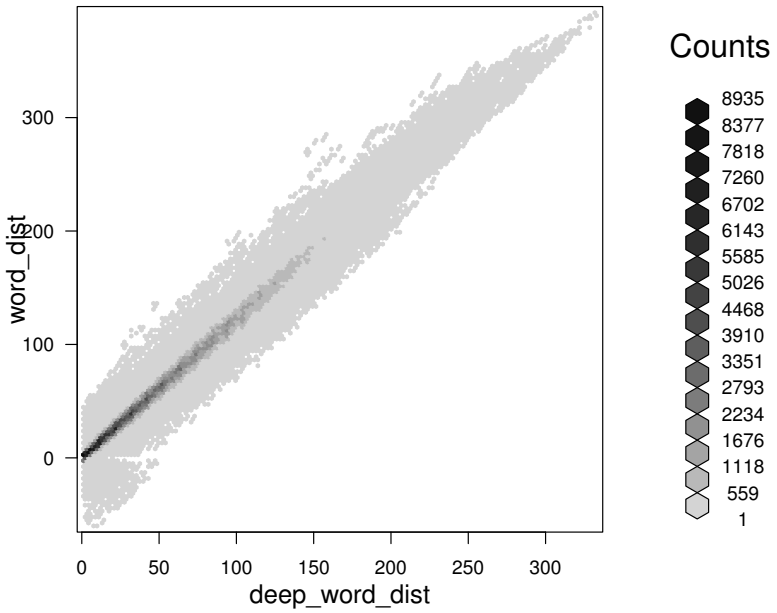


Figure 7.2: Correlation between non-quantized values of `deep_word_dist` and `word_dist` examined on the first 100 000 instances of training data

been altered three times in the way of omitting the former, the latter or both features. Result presented in Table 7.2 shows these facts:

Importance of any word distance information is pointed out by tiny differences between the models, which employed at least one of the compared features. On the other hand the model created with no explicit information about word distance achieved obviously worse performance.

Exclusive position of `word_dist` justifies the improvement of the model to F-score of 47.7% when using this feature alone[2] and of course by its easy way of retrieval.

Conclusions of the second point lead me to defining a new feature set `inter2_set`, which is the same as the `inter1_set` except the omitted `deep_word_dist` feature.

Feature set `inter2_set` served as a starting set for an experiment, which was done to show how features encapsulating information about distance contribute to the performance of the model. The same principle as for the experiments in the section 7.1.2 was used — features have been sequentially eliminated from the model.[3] Similarly to the analysis of equality and synonymy features (section 7.1.2), results of this experiment are depicted in Figure 7.3 and here I present some observation I have noticed:

[2]It means alone in the context of word distance features.

[3]However, the criteria of the removal order differed. As the first ones the features de-

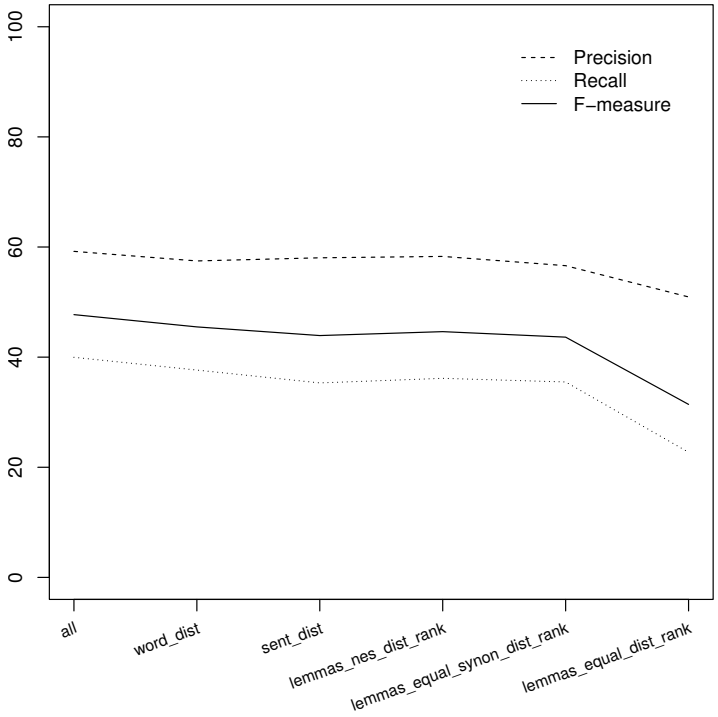


Figure 7.3: Performance changes when sequentially removing distance features from the model `inter2_set`

Relevancy of distance features is illustrated by slow decrease followed by a steep fall of over 12% after removal of the last distance related feature. This behavior is very similar to that presented for equality and synonymy features. On the other hand, the contribution of such features is not as significant as of those presented in the section 7.1.2, because success rate of the model without any distance feature still reaches 31.4% compared to the poor result of 4.1% without equality and synonymy features.

Ranking features, although they affect just the subset of instances, carry a crucial part of distance information. It means that distance features has a marginal effect on those instances whose participants do not match with any of the ranking condition.[4]

Improvement after NE feature removal could be caused by that the presence of the feature `lemmas_nes_dist_rank` coerced the model into less general behavior, what can be seen especially on increased value of recall.

scribing just the distance were removed following by ranking features where the distance information is combined with another one. Ranking features were ordered from the most specific to the most generic to show that the different order of their removal do not influence the decreasing trend substantially.

[4]Lemmas equality, synonymy, named entity type matching etc.

	Precision	Recall	F-measure
numbers_equal + genders_equal + negs_equal	59.21	39.97	47.72
both_numbers + both_genders + both_negs	58.97	39.80	47.52
all	58.87	39.64	47.37

Table 7.3: Comparison of equality and combination grammatical features in the model inter2_set.

7.1.4 Effect of grammatical features

I have incorporated several grammatical features which originate either from morphological or tectogrammatical layer of PDT. The latter layer contains grammatemes which are just tectogrammatical equivalents of morphological categories and as I already showed in Figure 5.1, they strongly correlate. This correlation was confirmed also during intermediate experiments, when they influenced the results really marginally, what resulted in fact that they are missing from the final feature set.[5]Hence I did not attempt to conduct a special analysis of the contribution of these features on the model. This fact was also a reason I did not enrich the feature set with the combination and equality of participants' grammatemes.

On the contrary, combination and equality features was indeed created for morphological categories. In the next experiment I focus on these features to find out, which ones are better. This experiment as well as the following use the model inter2_set as a source. Its results are presented in Table 7.3, from which we can conclude the following fact:

Equality plays slightly bigger role than combination. However, if we take into account the size of the data used, the difference is tiny and could be insignificant.

Following experiment is based on exactly the same approach as the experiments in sections 7.1.2 and 7.1.3. I examined a model dependency on grammatical features and in Figure 7.4 I present the results. Furthermore here are the observations:

Grammatical features are less influential compared to equality (synonymy) and distance features. After elimination of every grammatical feature, the model strongly outperforms[6] the models deformed by eliminations of features in sections 7.1.2 and 7.1.3.

[5]One can argue that they are part of the final model presented in the section 7.2. However, only those related to the anaphor are present, so the reason of their inclusion is different (see the section 7.1.5).

[6]Over 10% in case of missing distance features and over 37% in case of eliminated equality and synonymy features.

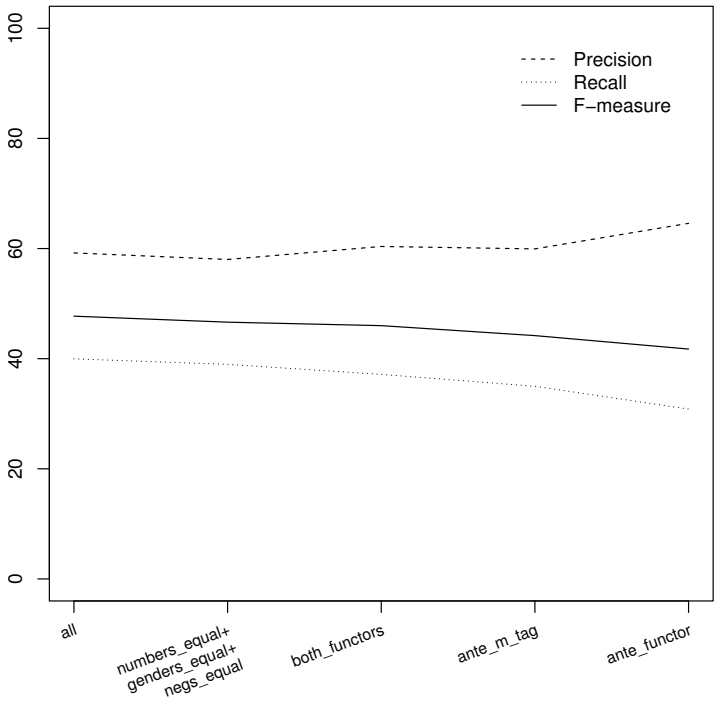


Figure 7.4: Performance changes when sequentially removing equality and synonymy features from the model `inter2_set`

Lack of functor information increases precision. Particularly the removal of `ante_functor` feature is exemplary, when the difference between precision and recall raises from almost 25% to almost 35%.

7.1.5 Effect of anaphor

Instead of features I analyzed in the sections 7.1.2 and 7.1.3, which are exclusively binary,[7] grammatical features include also unary ones, those which are defined just for anaphor or antecedent candidate.[8] Whereas antecedent unary features are useful for comparing all the antecedents belonging to a bundle, the asset of an anaphor unary feature is in dispute. It seems to be redundant, because for each instance in a bundle it has the same value.

To prove if this claim is correct, I conducted following experiment. Again it had started with the model `inter2_set` and afterwards continued not with eliminating, but adding of features. So the last model in a sequence contains all the features added to previous models. Figure 7.5 depicts the contribution of anaphor features to the model. In addition, I have noticed following interesting facts about it:

[7] A binary feature combines information provided by basic instances of both anaphor and antecedent candidate (more in Chapter 5).

[8] The name of unary feature always starts with either "anaph" or "ante".

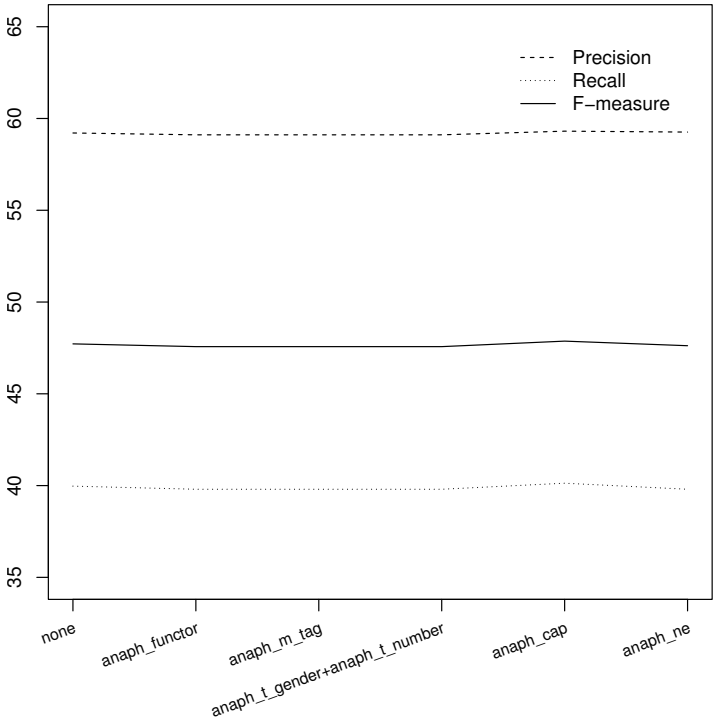


Figure 7.5: Performance changes when sequentially adding anaphor features to the model `inter2_set`

Bigger amount of anaphor features has a tiny but positive effect on the model whereas incorporating just some features do not change or lowers the F-score indeed. It is very interesting, because the only thing anaphor unary features can determine is the probability that the anaphor candidate is non-anaphoric. They cannot influence the choice of antecedent, if the instance is anaphoric. It looks like that in bigger number they are strong enough to correctly influence some decisions about anaphoricity. This can be illustrated by results of another experiment on the model `inter2_set`. After enriching the model only with the feature `anaph_cap`, the feature that caused the improvement in the previous experiment, the success rate remains the same.

In conclusion, inclusion of some of the anaphor features resulted in improvement of the model, quantified on reduced development data to F-score of 47.9%. Thus I introduce here this new feature set — `inter3_set` consisted of following features:

	Precision	Recall	F-measure
all	59.31	40.13	47.87
pw_p + wp_p	59.71	40.30	48.12
ante_rank_in_doc	59.95	39.97	47.96
all capital and gold NE features	56.57	39.97	46.84

Table 7.4: Contribution of the lexical features not analyzed until now during their sequential removal from the inter3_set. The feature set made after omitting the meronymy features is the final set.

anaph_functor, ante_functor, anaph_m_tag, ante_m_tag, anaph_t_gender, anaph_t_number, word_dist, sent_dist, ante_rank_in_doc, pw_p, wp_p, lemmas_equal_dist_rank, lemmas_equal_synon, lemmas_equal_synon_dist_rank, both_functors, numbers_equal, genders_equal, negs_equal, anaph_cap, ante_cap, lemmas_equal_both_cap, ante_ne, lemmas_nes_dist_rank

7.1.6 Effect of other lexical features

Except the lemmas' equality and synonymy features, there are three more types of features derived from lexical information: meronymy features, those concerning frequency of the lexical item in a document and those related with named entities. All three groups are included in every intermediate model presented here and in the final model, too.

Though meronymy/holonymy features seem to play a much more useful role in models of part-whole/whole-part bridging anaphora than in coreferential models, they have been included in many intermediate models during development of coreference resolver. Features pw_p and wp_p persisted in the models maybe by mistake or just because in particular stage[9] of development they increased the success rate of the model.

Therefore I again carried out a test, whether the model benefited from presence of meronymy features or not. The inter3_set model with the features pw_p and wp_p being excluded, performed better and exceeded the F-score of 48%. This became the final model final_set.

To show the contribution of the remaining groups of features I designed two experiments. In the first one the ante_rank_in_doc was eliminated from the final_set model and in the second one almost[10] all the features related to capitals or named entities were filtered out. The results presented in Table 7.4 points out the following fact:

[9] Probably in the initial one.

[10] Only anaph_cap was not removed. Since it is an anaphor feature, in companion with

	Precision	Recall	F-measure
lemmas_equal_both_cap + lemmas_nes_dist_rank	59.71	40.30	48.12
lemmas_equal_nes_equal + lemmas_nes_dist_rank	59.41	40.30	48.02
lemmas_caps_dist_rank + lemmas_equal_nes_equal	59.41	40.30	48.02
lemmas_equal_both_cap + lemmas_caps_dist_rank	58.65	40.46	47.89
lemmas_equal_both_cap + lemmas_equal_nes_equal	58.21	39.97	47.39
lemmas_caps_dist_rank + lemmas_nes_dist_rank	59.41	40.30	48.02

Table 7.5: Contribution of various combinations of gold NE and capital features utilizing equality or ranking on the `final_set` model.

Remaining lexical features contribute less than the features analyzed in the sections 7.1.2 and 7.1.3. While removal of named entity features decreased the f-score by 1.3%, exclusion of the feature related to frequency of a word in a document diminished the success rate merely by 0.1%.

Furthermore, I decided to examine how the success rate is changing when modifying the combination of named entity features. In the final model, following features are employed:

- capital features — `anaph_cap`, `ante_cap`, `lemmas_equal_both_cap`
- gold NE features — `ante_ne`, `lemmas_nes_dist_rank`

Presence of the anaphor unary feature, which is utilized because of reasons explained in section 7.1.5, and antecedent unary features, which are both incorporated, is understandable. It cannot be said about two remaining named entity features in the model, the combined Boolean feature from the first group and ranking feature from the second group. Therefore I conducted an experiment, where I replaced these two features with their unused counterparts in various combinations. The observation (see Table 7.5) is following:

Surprisingly, all the other combinations performed worse. In most of the combinations, the difference is very small. Except the combination without any ranking feature, I cannot give an explanation of the interactions, which can be just a work of chance.

7.2 Final models

After a lot of experiments that had been launched with just grammatical features, later followed by testing distance and lexical features, I came to the

other features of the same kind it improves the model and from the same reason it does not influence a selection of antecedent (see the section 7.1.5).

	Precision	Recall	F-measure
type0 coreference	59.7	40.3	48.1
NR coreference	20.0	0.9	1.8
PART_WHOLE bridging	0.0	0.0	0.0
WHOLE_PART bridging	0.0	0.0	0.0

Table 7.6: Success rates of final models trained on reduced training data and tested on reduced development data

model with the best performance among the plenty of other intermediate models I have developed.

The feature set that worked best on the reduced development data of type0 coreference consist of these features:

anaph_functor, ante_functor, anaph_m_tag, ante_m_tag, anaph_t_gender, anaph_t_number, word_dist, sent_dist, ante_rank_in_doc, lemmas_equal_dist_rank, lemmas_equal_synon, lemmas_equal_synon_dist_rank, both_functors, numbers_equal, genders_equal, negs_equal, anaph_cap, ante_cap, lemmas_equal_both_cap, ante_ne, lemmas_nes_dist_rank

Allowing just the features from the final_set feature set I constructed models for particular types of anaphora, this work is concerned in. I did not search for a feature set tailored to NR coreference, because coreference with specific reference and generic reference are very similar in terms of presented features.[11] In case of two tested bridging relations, after several experiments with various feature sets resulting in no non-zero success rate, I decided not to proceed with experiments employing another feature sets. However, the final tests were conducted and the final_set extended with pw_p and wp_p (say final_set_bridging) served as a final feature set for bridging relations.

Numbers that models trained from reduced training data achieve on reduced development data are presented in Table 7.6.

7.3 Evaluation tests

In section 7.2 I have found the feature set with the best performance in resolution of type0 coreference — final_set. I used this set to create a model for NR coreference as well and in slightly modified version[12] for bridging relations. However, all the models were due to performance issues trained and tested on the reduced data. Moreover these models were tested only on the development

[11] It is confirmed by the error analysis described in the section 7.5.

[12] Features pw_p and wp_p added.

Model	Development data			Evaluation data			Evaluation data approximation		
	P	R	F	P	R	F	P	R	F
type0.baseline	27.14	58.39	37.06	28.07	59.53	38.15	26.89	54.45	36.00
type0	62.26	30.44	40.89	65.49	30.95	42.03	54.70	28.31	37.31
type0.small	54.62	36.68	43.89	55.42	36.27	**43.84**	48.51	33.17	**39.40**
NR.baseline	9.84	62.93	17.02	9.49	62.06	**16.47**	9.26	53.41	**15.79**
NR	27.78	0.42	0.83	46.15	0.64	1.27	3.66	0.55	0.96
NR.small	22.58	0.59	1.15	25.00	0.75	1.46	3.91	0.65	1.11
PART_WHOLE	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
WHOLE_PART	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 7.7: Final results of individual models on the complete development and evaluation data. On the right-hand side there is the unbiased approximation of success rates.

data. Now it is time to create models from the complete data and test them on the complete development data. In addition, thus far the evaluation testing data have not been used, but this is the moment they come on the scene to ensure reliable results not biased by knowledge of the data.[13]

I tested models trained from all training data (type0, NR), models trained from the reduced training data (type0.small, NR.small) and I also tested baseline approaches chosen in the section 6.2 (type0.baseline, NR.baseline).

Overall results that are tabulated in Table 7.7 are a source for interesting observations and conclusions:

type0 coreference models are better than baselines. Their performance gain originates mainly from the higher precision values. On the other hand machine learning models of NR coreference absolutely failed. Their poor recall shows that they cover very few coreference occurrences. In my opinion, the relevancy of features to describe the coreference is in both type very similar. However, since type0 coreference is three times more frequent, all these features are in process of creating the model of NR penalized, because they describe all mentions, which are non-coreferential or coreferential with another type, better. If some distinctive feature was involved, it would be given bigger weights and recall would probably increase. The results of bridging relations resolution was unfortunately at zero.

Reduced models reach lower precision and higher recall than those trained from the complete data. Larger models cause that the model is more precisely trained to resolve well-distinguishable anaphora at the expense of the others, not so clear ones, which results in recall decrease.

[13] As they are in the case of development data, because during development they are analyzed and to achieve better results on them is the purpose of model modification.

The evaluation data perform better than the development data. Since the proportion of anaphoric mentions is almost the same (see Table 3.2), the difference could come from the higher proportion of easily resolvable anaphora. This claim also gives an explanation why the model `type0.small`, which is, according to conclusions of previous point, less precise in resolution of clear coreferences, is the only case where the F-score on development data is slightly higher.

Substantial loss of `type0.small` model compared to the results achieved on the reduced development data (in Table 7.6) is obvious. It may have the same origin as in previous case. Furthermore, the overall proportion of `type0` coreference[14] might have influenced the result.

In the section 4.2 I mentioned that the instances in a bundle were constructed by the combination of basic instances belonging to the anaphor candidate and all the antecedent candidates lying in the pre-defined sentence window. According to the section 4.6 the size of the window was 4 previous sentences for bridging relations and 10 previous sentences for coreference[15] and it was applied on all types of data. However, constraining the testing data in this way artificially improves the success rate because the resolver is not penalized, if it marks the anaphor, whose anaphoric link targets outside the window, as non-anaphoric.

To gain the results which are not biased by this constraint, I introduce the approximated success rates whose calculation I illustrate on the example of results of `type0.small` model performed on the evaluation data. The precision and recall values originate from the following counts:

$$R = \frac{\#correctly\_predicted\_positive}{\#all\_positive} = \frac{1043}{2876} = 36.27$$
$$P = \frac{\#correctly\_predicted\_positive}{\#predicted\_positive} = \frac{1043}{1882} = 55.42$$

Thus there are 2876 true links considering the window of size 10. From the distribution of coreference distances measured on the eval data follows that the number of links inside the window accounts for $window\_cov = 91.5\%$.[16] So without constraint on the window size, the evaluation data contains 3144 positive bundles. Then the recall approximation on unconstrained data is calculated with this number of positive bundles (Equation 7.1). The worst case in precision is when all the links pointing outside the window were originally predicted as non-anaphoric, which was according to the constrained data correct.

[14] 14,7%, opposed to 13,7% in the complete development data (see Table 3.2).

[15] Always including the current sentence.

[16] The distribution should be similar to that on the training data illustrated in Figure 4.1. However, the coverage on the evaluation data is lower compared to 94.3% on the training data.

In the unconstrained data they all are predicted as non-anaphoric mistakenly. The precision approximation has to take this into account (Equation 7.2). The F-measure approximation is finally calculated in a standard manner from the precision and recall approximations.

$$R_{approx} = \frac{\#correctly\_predicted\_positive}{\frac{\#all\_positive}{window\_cov}} = \frac{1043}{3144} = 33.17 \quad (7.1)$$

$$P_{approx} = \frac{\#correctly\_predicted\_positive}{\#predicted\_positive + (\#all\_positive - \frac{\#all\_positive}{window\_cov})} \quad (7.2)$$

$$= \frac{1043}{2150} = 48.51$$

$$F_{approx} = 39.40 \quad (7.3)$$

I present the calculated success rate approximations for all evaluation results in Table 7.7. These results are unbiased by the window size constraint, hence these are the results, which should be used for comparisons with another works.

7.4 Significance of results

When conducting statistical experiments we should not be satisfied with one-value results. The value could be an outlier and the real value, we would like to estimate, may be different. A confidence interval is the solution. It offers a reliable estimate of interval, in which the real parameter is included with the exactly defined probability. I search for both-sided 95% confidence interval of F-scores on evaluation data. It means that the interval covers the real F-score with probability 95% and the outliers can lie on the both sides of the interval.

Several methods to calculate a confidence interval exist. One of them is to calculate it on the basis of quantiles of Gaussian distribution. However, I utilize another one — the bootstrapping method Venables et al. [2002]. Bootstrapping lies in multiple (here 1000 times) sampling from the testing data. Samples are of the same size as the data, thus they must be sampled with replacement. On every sample the statistics (F-score) is calculated. Then the calculated values are sorted and 2.5% (25 in this case) of values from the beginning and same amount from the end of the list is removed. The first and the last value in the list that remained form the requested confidence interval.

I applied this method on evaluation data for models that solved the particular tasks most successfully[17] and I was given the intervals in Table 7.8. These intervals stand for reliable estimates of real success rate of resolver, again with approximated interval for evaluation data without window size constraint. It confirms that the F-score of 48.1% of the `type0.small` model on reduced development data is an outlier.

[17] From understandable reasons I omitted `PART_WHOLE` and `WHOLE_PART` models.

Model	Confidence interval (95%)	Confidence interval (95%) approximation
`type0.small`	(42.06% ; 45.68%)	(37.85% ; 41.15%)
`NR.baseline`	(0.60% ; 2.61%)	(14.66% ; 16.97%)
`type0.small−type0`	(0.73% ; 2.97%)	(1.17% ; 3.07%)
`type0−type0.baseline`	(2.10% ; 5.58%)	(-0.20% ; 2.88%)

Table 7.8: Confidence intervals of performance and difference in performance (in F-score) on evaluation data.

Furthermore I would like to find out, whether the three presented models of `type0` coreference are significantly of different quality as the one-value results indicate. To examine this I again employed the bootstrapping method though with a small modification. Instead of F-score, the statistics was calculated as a subtraction of F-scores belonging to two selected models. If such constructed interval covers 0, the difference between the models is statistically insignificant, otherwise one model is significantly better then the other one. Which one is better depends on whether the interval is positive or negative.

In this way I carried out comparison of the `type0.small` model with the `type0` model and the `type0` model with the `type0.baseline`. According to the approximated results in Table 7.8 I conclude the following relation between the models:

$$\texttt{type0.baseline} \approx \texttt{type0} < \texttt{type0.small} \tag{7.4}$$

Symbols $\approx$ and $<$ stand for the relation of ordering on F-scores of models gained on evaluation data. Although the model `type0` is significantly better than `type0.baseline` when considering only the antecedents in the window of size 10, the approximation for all true antecedents penalizes the model with higher precision more, which results in no significant difference between these two models.

7.5 Error analysis

In the end I conducted an error analysis in order to find out what were the typical errors that my resolver had committed. Whether regarding the bridging relations I obtained merely zero results, I carried out the analysis just on coreference. I randomly selected 20 erroneous bundles from the development data.

In case of `coref0` coreference, there were 15 errors when the mention was anaphoric and the resolver said it was not, 3 errors when the resolver labeled another candidate as the antecedent and finally 2 errors, when the mention was not anaphoric but the resolver marked an antecedent for it.

Moreover, I classified errors according to their lexical and semantic properties. There were 5 errors when the whole phrase of the anaphor was equal to the whole phrase of the true antecedent. In 2 of them, the phrases were made of just one node on the t-layer,[18] for example:

(7.5)
v pondělí	←	do pondělí
on Monday	←	until Monday

Since the second error was also on common name,[19] this problem possibly arises because of training the common and proper name anaphors together into a joint model. Then the weight for equality of common nouns is lower, so they are less frequently marked as coreferential.

In the 3 remaining cases at least one of the mentions had more than one node on the t-layer, for instance:

(7.6)
Lidové noviny	←	Lidové noviny
People's newspaper	←	People's newspaper

Moreover there are two cases when the heads of the mentions were equal but their subtrees differed and the mentions were mistakenly resolved as coreferential, as it is illustrated in Figure 7.6. From these cases one can see that enriching the model by the features based on the whole mention equality could improve the results. But this is not so clear as it seems to be. Another two cases, where the heads of the mentions were equal but their subtrees differed, were coreferential but were not resolved so, for example:

(7.7)
skok z okna bytu v prvním poschodí ve Vlkově ulici v Brně	←	skok
the jump from the window of the flat on the first floor in Vlkova Street in Brno	←	the jump

Ten errors occurred when a more complex lexical relation between the anaphor and the true antecedent was not discovered. In two of them, the relation could be resolved if the weight of synonymy between the heads were higher, for instance in case:

(7.8)
resort	←	ministerstvo
department	←	ministry

But this would not work in two another cases of those 10, where the phrases were synonymous but the heads alone not. Another five errors from those 10 were on the anaphors, whose true antecedent's head was hyperonymous or hyponymous with the anaphor's head. In Figure 7.7 there is one example of hyperonymy between the heads. However, if the relation of hyperonymy or

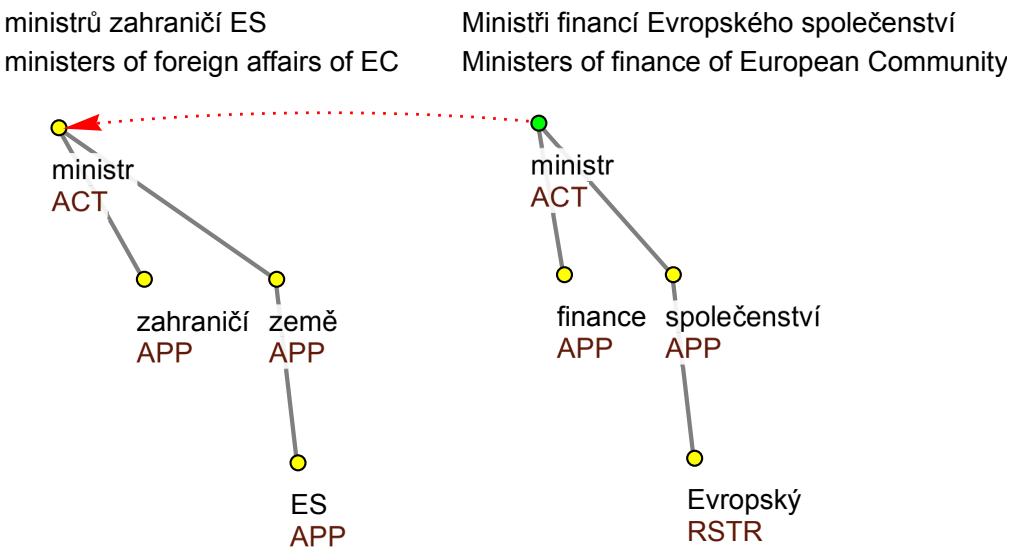


Figure 7.6: Erroneously labeled coreference caused by comparing just heads.

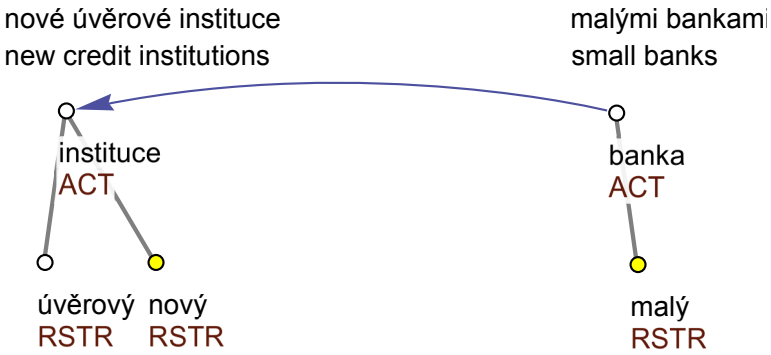


Figure 7.7: Not resolved coreference, where the mention heads are in relation of hyperonymy.

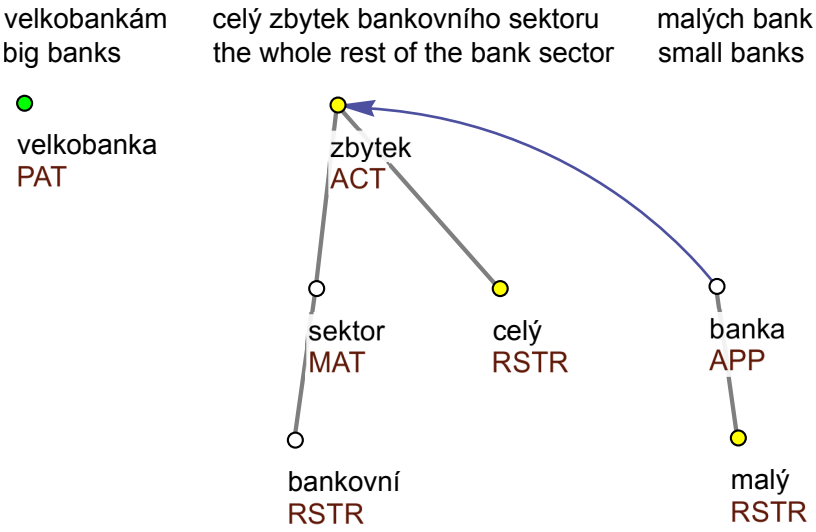


Figure 7.8: Hardly resolvable coreference. The mentions are coreferential only in the context of the third mention ("big banks").

synonymy between the heads was covered perfectly, this could not solve the problems of distinctions caused by the descendants.

The last case of those ten is very interesting. It is depicted in Figure 7.8 and one can see, that the synonymy between the mentions is valid only in context of the third word contrastive to the anaphor.

The remaining case, presented in Figure 7.9, should not be marked as an error in truth. The real antecedent is an apposition consisting of two phrases. One of them is correctly predicted as being the antecedent. Nevertheless, according to the annotation rules Mikulová et al. [2006], the apposition is handled by a special technical node and the coreferential link targets to this node. Since the resolver described in this work filters only the noun antecedents, following the rule described in the section 4.2 the coreferential chain is traversed until the first noun is found. So even the apposition node is not the true antecedent in this case. This bug can be easily fixed by extending the antecedent filter also to apposition nodes together with the postprocessing, which redirects the link that refers to the member of apposition to the apposition node itself.

In spite of the fact that the best model for `NR` coreference is the baseline model, I did not conduct the error analysis on the results output by this model, because due to its simple construction using the single feature it is easily predictable what errors it committed. Therefore I randomly selected 20 erroneous bundles from the development data after being processed by `NR.small` model.

The error analysis confirmed what the tiny value of recall on both development and evaluation data indicated. All 20 cases were in fact coreferential although

[18]Prepositions are not represented on t-layer as a node Mikulová et al., 2006.
[19]In Czech "Monday" is a common name and starts with lower-case — "pondělí".

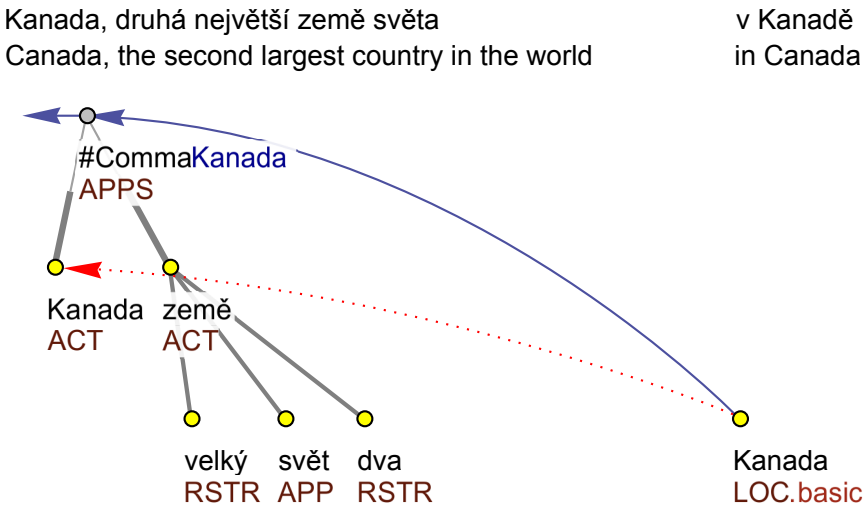


Figure 7.9: Erroneously labeled coreference because of not handling with the annotation rules for apposition.

the resolver claimed they were non-anaphoric. Also my hypothesis about reasons of this behavior confirmed. It happens because the `NR` shares the key features with `coref0` coreference, but in case of the `NR` coreference they are penalized only because `NR` type is less frequent. It justifies the observation that in the majority — 17 of the erroneous cases the heads of anaphor and true antecedent were identical. Moreover, if I try to resolve these error cases using the `coref0.small` model, it yields 16 correct assignments with the F-score of 84%.

Chapter 8

Conclusion

The goal of this work was to make the first step in the area of extended anaphora on Czech language data. It mainly concentrated on the noun phrase coreference, identity-of-sense anaphora and partially on bridging relations. This advancement became possible only thanks to the project of annotation the noun phrase coreference and bridging relations in the Prague Dependency Treebank 2.0, which provided me with the data necessary for training and testing.

The resolver proposed in this work employed some of the state-of-the-art approaches in the task of anaphora resolution like ranking as well as the joint model for anaphoricity determination and anaphora link resolution. The model trained by the resolver incorporated mainly the lexical and distance features, which proved to carry the most valuable information for the noun phrase coreference resolution. On this type of anaphora the resolver achieved the F-score of 39.4%.

However, the room for improvement in this area of research is still wide enough. The error analysis showed that information describing the other words in the mention than the head could improve the success rate. I have not attempted to use the information from the analytical layer of PDT. The analytical functor, especially, could influence the model positively. In this work I acquired the shallowly annotated corpora — Czech National Corpus to approximate the meronymy relations. Although it brought no performance improvement, I see the potential in the sensible exploiting of the information from the large unannotated data, for example by utilizing the various association measures between the words. Concerning the design of resolver, employing the separate models for proper noun and common noun anaphor could help as well. In the case of identity-of-sense anaphora, the main task for the future is to find the features, which differentiate it from the identity-of-reference anaphora.

Bibliography

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalová, Marius Pasca, and Aitor Soroa. A study on similarity and relatedness using distributional and wordnet-based approaches. In *HLT-NAACL*, pages 19–27, 2009. 5.3.3

Amit Bagga and Breck Baldwin. Entity-based cross-document coreferencing using the vector space model. In *COLING-ACL*, pages 79–85, 1998. 2.4, 6.1

Adam Berger, Vincent Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing, 1996. 4.5, 4.5, 5.3.3

Ondřej Bojar, Zdeněk Žabokrtský, Miroslav Janíček, Václav Klimeš, Jana Kravalová, David Mareček, Václav Novák, Martin Popel, and Jan Ptáček. Czeng 0.9, 2009. 5.3.3

Eugene Charniak and Micha Elsner. EM works for pronoun anaphora resolution. In *EACL*, pages 148–156, 2009. 2

CNC. Czech national corpus – SYN2005, 2005. 5.3.3

Pascal Denis and Jason Baldridge. A ranking approach to pronoun resolution. In *IJCAI*, pages 1588–1593, 2007a. 2.1, 4.5

Pascal Denis and Jason Baldridge. Joint determination of anaphoricity and coreference resolution using integer programming. In *HLT-NAACL*, pages 236–243, 2007b. 2.2

Pascal Denis and Jason Baldridge. Specialized models and ranking for coreference resolution. In *EMNLP*, pages 660–669, 2008. 2.1, 2.3, 2.4, 5.3.6

Aria Haghighi and Dan Klein. Simple coreference resolution with rich syntactic and semantic features. In *EMNLP*, pages 1152–1161, 2009. 2.4

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, Zdeněk Žabokrtský, and Magda Ševčíková-Razímová. Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia, 2006. 3.1

Jiří Hana, Daniel Zeman, Jan Hajič, Hana Hanová, Barbora Hladká, and Emil Jeřábek. Manual for morphological annotation, revision for the Prague

Dependency Treebank 2.0. Technical Report TR-2005-27, ÚFAL MFF UK, Praha, Czechia, 2005. 5.3.1, 5.3.6

Lucie Kučová and Eva Hajičová. Coreferential relations in the Prague Dependency Treebank. In *Proceedings of DAARC2004*, pages 97–102, 2004. 1.2

Xiaoqiang Luo. On coreference resolution performance metrics. In *HLT/EMNLP*, 2005. 2.4, 6.1

Marie Mikulová, Alevtina Bémová, Jan Hajič, Eva Hajičová, Jiří Havelka, Veronika Kolářová, Lucie Kučová, Markéta Lopatková, Petr Pajas, Jarmila Panevová, Magda Razímová, Petr Sgall, Jan Štěpánek, Zdeňka Urešová, Kateřina Veselá, and Zdeněk Žabokrtský. Annotation on the tectogrammatical level in the Prague Dependency Treebank. Annotation manual. Technical Report 30, ÚFAL MFF UK, Prague, Czech Rep., 2006. 5.2, 1, 7.5, 18

Ruslan Mitkov. *Anaphora Resolution*. Longman, London, 2002. 1.2, 2

MUC-6. Coreference task definition. In *Proceedings of the Sixth Message Understanding Conference*, San Francisco, CA, 1995. Morgan Kaufmann. 2.4

MUC-7. Coreference task definition. In *Proceedings of the Seventh Message Understanding Conference*, San Francisco, CA, 1998. Morgan Kaufmann. 2.4

Vincent Ng. Unsupervised models for coreference resolution. In *EMNLP*, pages 640–649, 2008. 2, 2.4

Vincent Ng. Graph-cut-based anaphoricity determination for coreference resolution. In *HLT-NAACL*, pages 575–583, 2009. 2.2, 2.4

Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In *ACL*, pages 104–111, 2002. 2.1, 2.2, 2.4

Giang Linh Nguy. Návrh souboru pravidel pro analýzu anafor v českém jazyce. Master's thesis, MFF UK, Prague, Czech Republic, 2006. In Czech. 1.1, 2.3, 2.4

Giang Linh Nguy and Zdeněk Žabokrtský. Rule-based approach to pronominal anaphora resolution applied on the Prague Dependency Treebank 2.0 data. In *Proceedings of the 6th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2007)*, pages 77–81, 2007. 2.4

Giang Linh Nguy, Václav Novák, and Zdeněk Žabokrtský. Comparison of classification and ranking approaches to pronominal anaphora resolution in Czech. In *Proceedings of the SIGDIAL 2009 Conference*, pages 276–285, London, UK, September 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W09-3939. 2.1, 2.4, 6.2

NIST. ACE evaluation plan. Technical report, 2007. URL http://www.itl.nist.gov/iad/mig/tests/ace/2007/. 2.4

Anja Nědolužko. *Zpracování rozšířené textové koreference a asociační anafory na tektogramatické rovině v Pražském závislostním korpusu*. PhD thesis, MFF UK, Praha, Czech Republic, 2009. In Czech. 1.2, 3, 1.2, 3.2, 3.2

Anja Nědolužko, Jiří Mírovský, Radek Ocelák, and Jiří Pergler. Extended coreferential relations and bridging anaphora in the Prague Dependency Treebank. In *Proceedings of the 7th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2009)*, 2009. (document), 9, 3.2, 3.2, 3, 6.3, 6.2

Václav Němčík. Anaphora resolution. Master's thesis, FI MU, Brno, Czech Republic, 2006. In English. 2.4

Franz Josef Och, Christoph Tillmann, Hermann Ney, and Lehrstuhl Für Informatik. Improved alignment models for statistical machine translation. In *University of Maryland, College Park, MD*, pages 20–28, 1999. 5.3.3

Petr Pajas and Jan Štěpánek. XML-based representation of multi-layered annotation in the PDT 2.0. In *Proceedings of the LREC Workshop on Merging and Layering Linguistic Information (LREC 2006)*, pages 40–47, 2006. 4

Karel Pala and Jan Všianský. *Slovník českých synonym*. Nakladatelství Lidové noviny, 2000. 2

Massimo Poesio, Tomonori Ishikawa, Sabine Schulte im Walde, and Renata Vieira. Acquiring lexical knowledge for anaphora resolution. In *In Proceedings of the 3rd Conference on Language Resources and Evaluation (LREC)*, pages 1220–1224, 2002. 5.3.4

Massimo Poesio, Rahul Mehta, Axel Maroudas, and Janet Hitzeman. Learning to resolve bridging references. In *ACL*, pages 143–150, 2004. 2.3

Md. Altaf Rahman and Vincent Ng. Supervised models for coreference resolution. In *EMNLP*, pages 968–977, 2009. 2.1, 2.2, 2.4

Wee Meng Soon, Hwee Tou Ng, and Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544, 2001. 2.1, 2.2

Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. 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 656–664, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P09/P09-1074. 2.3, 4.2, 5.3.6

W. N. Venables, Brian D. Ripley, and W. N. Venables. *Modern applied statistics with S*. Springer, New York, 4th ed edition, 2002. 7.4

Marc B. Vilain, John D. Burger, John S. Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *MUC*, pages 45–52, 1995. 2.4, 6.1

Piek Vossen, editor. *EuroWordNet: a multilingual database with lexical semantic networks*. Kluwer Academic Publishers, Norwell, MA, USA, 1998. ISBN 0-7923-5295-5. 5.3.4

Xiaofeng Yang, Guodong Zhou, Jian Su, and Chew Lim Tan. Coreference resolution using competition learning approach. In *ACL*, pages 176–183, 2003. 2.1, 2.4

Appendix A

Feature weights

In this appendix I present the features and their values contained in the `final_set` feature set. For the purpose of fitting the requirements of maximum entropy algorithm the features were transformed to binary-valued ones, so that each value had its own feature. During the training, the learning algorithm assigned each feature a weight. The weights of features, which I also present in this appendix, correspond to the `type0.small` model, the best model for the noun phrase coreference resolution introduced in this work.

Even though there are several features that contain many values, e.g. the `both_functors` feature with 2079 different values, I do not depict all the values. Concerning the features of size more than 36 values, only the 18 values with the highest and 18 with the lowest weight are tabulated. The values are sorted by their weights in descending order.

Feature `anaph_t_gender`

This feature contains following 6 values:

Value	Weight	Value	Weight	Value	Weight
fem	2.59062e-08	anim	3.37536e-09	__undef__	-3.57114e-11
inan	1.55507e-08	nr	-7.09003e-12	neut	-8.97595e-10

Feature `anaph_t_number`

This feature contains following 4 values:

Value	Weight	Value	Weight
sg	6.88131e-08	nr	4.00928e-10
pl	9.61603e-09	__undef__	-3.57114e-11

Feature anaph_functor

This feature contains 52 values. The following table shows the 18 ones with the highest weight:

Value	Weight	Value	Weight	Value	Weight
PAT	1.10129e-08	DENOM	2.63661e-12	EXT	1.37557e-12
ACT	7.28201e-09	COMPL	2.53457e-12	CPR	1.34439e-12
TWHEN	3.20072e-10	TSIN	2.20238e-12	DIR2	1.2179e-12
DIR3	4.09172e-11	TTILL	2.08126e-12	TPAR	1.08864e-12
ID	1.34834e-11	ORIG	1.94802e-12	RESTR	1.03902e-12
COND	3.85723e-12	DPHR	1.73932e-12	AUTH	7.48707e-13

The next table shows the 18 values with the lowest weight:

Value	Weight	Value	Weight	Value	Weight
MAT	-1.26673e-12	EFF	-2.11544e-11	ADDR	-5.08064e-11
CAUS	-7.39191e-12	MEANS	-2.23059e-11	DIR1	-5.86367e-11
MANN	-1.49485e-11	REG	-2.44746e-11	RSTR	-6.61214e-11
FPHR	-1.52772e-11	PAR	-2.71166e-11	ACMP	-7.22133e-11
AIM	-1.809e-11	CPHR	-3.60484e-11	LOC	-2.35899e-10
CRIT	-1.93753e-11	BEN	-4.05592e-11	APP	-1.82854e-09

Feature anaph_m_tag

This feature contains 105 values. The following table shows the 18 ones with the highest weight:

Value	Weight	Value	Weight
NNIS2-----A----	7.89071e-10	NNNS4-----A----	1.69834e-10
NNIS1-----A----	7.11867e-10	NNIS6-----A----	1.60073e-10
NNFS4-----A----	6.25437e-10	NNNS1-----A----	1.55207e-10
NNIS4-----A----	4.34939e-10	NNMS2-----A----	1.29967e-10
NNFP2-----A----	3.99941e-10	NNMS1-----A----	9.29073e-11
NNFS6-----A----	3.97195e-10	NNMP2-----A----	6.10696e-11
NNFS1-----A----	3.86071e-10	NNNS6-----A----	4.19393e-11
NNNS2-----A----	3.53556e-10	NNFS2-----A----	1.71208e-11
NNIP2-----A----	1.81685e-10	NNMP3-----A----	3.28736e-12

The next table shows the 18 values with the lowest weight:

Value	Weight	Value	Weight
NNXXX-----A---8	-8.99304e-12	NNNS7-----A----	-2.98386e-11
NNFS7-----A----	-9.2203e-12	NNFP6-----A----	-3.0375e-11
NNFP1-----A----	-9.52543e-12	NNIS7-----A----	-3.16649e-11
NNIS6-----A---1	-1.25801e-11	NNFP4-----A----	-3.76579e-11
NNMP4-----A----	-1.56203e-11	NNNP2-----A----	-3.82102e-11
NNMS7-----A----	-1.67884e-11	NNIP1-----A----	-3.90694e-11
NNIP6-----A----	-1.97564e-11	NNMP1-----A----	-4.93836e-11
NNFS3-----A----	-2.47753e-11	NNIP4-----A----	-5.2821e-11
NNMS4-----A----	-2.90751e-11	NNFXX-----A---8	-6.23054e-11

Feature ante_functor

This feature contains 53 values. The following table shows the 18 ones with the highest weight:

Value	Weight	Value	Weight	Value	Weight
ADDR	1.38683	DIR1	0.626517	RESTR	0.0729718
ACT	1.28626	__undef__	0.607846	MOD	-0.0831445
PAT	1.10766	SUBS	0.503556	ACMP	-0.110116
LOC	0.942271	DENOM	0.281375	HER	-0.123652
DIR2	0.923276	AUTH	0.105923	CM	-0.262227
APP	0.83923	PAR	0.0836932	BEN	-0.400023

The next table shows the 18 values with the lowest weight:

Value	Weight	Value	Weight	Value	Weight
TPAR	-1.28758	INTT	-1.54648	TFRWH	-2.64156
CNCS	-1.32327	COND	-1.5586	ATT	-2.74008
FPHR	-1.3481	TFHL	-1.55944	DPHR	-4.11911
RESL	-1.36213	EXT	-1.61196	TSIN	-4.33567
THL	-1.44036	RSTR	-1.68337	DIFF	-4.80642
TOWH	-1.47769	TTILL	-1.75551	COMPL	-5.28504

Feature ante_m_tag

This feature contains 106 values. The following table shows the 18 ones with the highest weight:

Value	Weight	Value	Weight
NNNS1-----N----	3.21073	NNFXX-----A---8	0.926034
NNMPX-----A---8	2.8872	NNNP7-----A----	0.920607
NNMPX-----A----	2.58391	NNFP6-----A---1	0.915438
NNMP6-----A----	1.57642	NNIS3-----A----	0.842988
NNMSX-----A----	1.24866	NNIXX-----A---8	0.791502
NNFPX-----A---8	1.15649	NNMS3-----A----	0.6833
NNFP1-----A---1	1.09856	NNMS6-----A----	0.68177
NNIPX-----A---8	1.07976	NNIS4-----A----	0.621899
NNNXX-----A---8	0.97259	NNFS2-----A---1	0.621893

The next table shows the 18 values with the lowest weight:

Value	Weight	Value	Weight
NNXXX-----A---8	-1.24494	NNFS2-----N----	-2.36738
NNIS7-----A---1	-1.34848	NNFS4-----N----	-2.47283
NNNP4-----A----	-1.50072	NNFS7-----A---1	-2.49502
NNMP1-----A---1	-1.63579	NNFS1-----N----	-2.93738
NNXXX-----A----	-2.03833	NNIS1-----A---1	-3.16368
NNIS3-----A---1	-2.16521	NNNP3-----A----	-3.45694
NNNP4-----A---2	-2.18887	NNIP2-----A---1	-4.08629
NNNS7-----A---8	-2.22904	NNMXX-----A----	-4.09909
NNMS6-----A---1	-2.34346	NNNS6-----A---1	-4.25508

Feature word_dist

This feature contains following 8 values:

Value	Weight	Value	Weight	Value	Weight
0..20	0.939358	60..120	-0.228328	__undef__	-0.591838
20..40	0.573982	120..180	-0.304389	<0	-9.31531
40..60	0.295831	>180	-0.468403		

Feature sent_dist

This feature contains following 12 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	3	0.136038	7	-0.34052
1	0.312608	6	0.0139924	8	-0.412145
2	0.25489	5	-0.0223894	10	-0.687577
4	0.224607	9	-0.212774	0	-0.83221

Feature ante_rank_in_doc

This feature contains following 25 values:

Value	Weight	Value	Weight	Value	Weight
22	0.985801	6	0.176412	17	-0.501435
__undef__	0.607846	9	0.161934	13	-0.553629
20	0.397837	23	0.13817	16	-0.642167
4	0.371667	8	0.0785547	18	-0.722692
2	0.359194	10	-0.091597	15	-0.76158
7	0.267568	14	-0.24888	21	-0.945578
3	0.241839	19	-0.371457	24	-2.29895
1	0.207179	11	-0.472614		
5	0.194396	12	-0.476629		

Feature lemmas_equal_dist_rank

This feature contains following 14 values:

Value	Weight	Value	Weight	Value	Weight
1	1.9956	12	-0.0471335	10	-2.5583
2	0.802017	13	-0.050762	8	-2.59737
5	0.597066	11	-0.302513	6	-4.36878
3	0.557451	4	-0.766836	7	-4.81432
__undef__	0.419492	9	-0.810891		

Feature lemmas_equal_synon

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	1	0.492758	0	-1.46801

Feature lemmas_equal_synon_dist_rank

This feature contains following 15 values:

Value	Weight	Value	Weight	Value	Weight
10	1.66508	5	-0.152477	13	-1.71914
7	0.989439	3	-0.158758	12	-1.94863
1	0.604492	6	-0.397245	11	-2.15871
2	0.396583	__undef__	-0.492758	9	-3.77333
4	0.122776	14	-0.943278	8	-4.80468

Feature both_functors

This feature contains 2079 values. The following table shows the 18 ones with the highest weight:

Value	Weight	Value	Weight	Value	Weight
VOCAT:ADDR	19.0721	CRIT:EFF	6.55182	DIR2:DIR1	5.9589
TPAR:THL	10.9167	CAUS:MANN	6.45643	APP:SUBS	5.91958
CPR:CPR	7.64267	ADDR:ORIG	6.44064	CPHR:EFF	5.81282
TWHEN:TPAR	7.12754	APP:THL	6.28793	ACT:TTILL	5.77114
APP:TOWH	6.94973	TPAR:TWHEN	6.10058	TWHEN:CPR	5.70535
LOC:COND	6.62693	DENOM:BEN	6.0097	MAT:DIR2	5.618

The next table shows the 18 values with the lowest weight:

Value	Weight	Value	Weight	Value	Weight
DIR2:ADDR	-4.52352	AIM:ACT	-4.80267	FPHR:APP	-5.40889
EFF:APP	-4.56634	COMPL:ACT	-4.81378	MANN:PAT	-5.4156
ID:RSTR	-4.60449	PAT:SUBS	-4.87346	FPHR:ACT	-5.85981
RSTR:BEN	-4.66991	MANN:ACT	-5.11545	ID:ORIG	-6.0182
BEN:DENOM	-4.68234	RSTR:TWHEN	-5.26559	ADDR:DENOM	-6.12941
FPHR:PAT	-4.75823	PAR:PAT	-5.29137	ID:APP	-8.21951

Feature numbers_equal

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	1	0.435551	0	-0.577684

Feature genders_equal

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	1	-0.123782	0	-0.873788

Feature negs_equal

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	1	-0.282116	0	-1.14629

Feature anaph_cap

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	1	-0.260365	0	-0.589408

Feature ante_cap

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	0	-0.454219	1	-0.5855

Feature lemmas_equal_both_cap

This feature contains following 3 values:

Value	Weight	Value	Weight	Value	Weight
__undef__	0.607846	0	-0.260923	1	-0.367539

Feature ante_ne

This feature contains following 15 values:

Value	Weight	Value	Weight	Value	Weight
GK	2.34578	KS	-0.163396	R	-0.846959
KR	1.71208	GS	-0.206291	Y	-1.22748
S	0.0541572	K	-0.402877	KY	-1.55271
G	-0.106811	__undef__	-0.431726	SY	-1.7729
E	-0.123151	H	-0.82141	U	-4.12187

Feature lemmas_nes_dist_rank

This feature contains following 14 values:

Value	Weight	Value	Weight	Value	Weight
6	6.28471	2	-0.0881074	7	-1.13741
1	1.32161	__undef__	-0.136651	10	-1.29758
3	0.343549	11	-0.302513	4	-3.58335
12	-0.0471335	9	-0.339815	5	-4.19179
13	-0.050762	8	-0.486598		

Appendix B

Content of the attached CD

`/` — main directory contains all the scripts necessary in the process of resolution and evaluation

`/utils` — utility scripts not used in the main resolution process

`/lib` — Perl modules used in the scripts

`/prereq` — this directory contains the installation packs of all prerequisites that the resolver requires

`/queries` — queries on corpora to retrieve the data from them

`/data/pdt` — original PDT documents

`/data/source_lists` — partitioning of the PDT documents into training and testing data

`/data/extracted` — simple tables extracted from the PDT documents using btred in the first sub-stage of preprocessing

`/data/tables/original` — the output tables after the second sub-stage of preprocessing are generated here (because of their enormous size, it is empty now)

`/data/tables/extended` — the output tables after the third sub-stage of preprocessing are generated here (because of their enormous size, it is empty now)

`/data/tables` — the tables which serves as an input to TADM resolver

`/model` — trained models are stored here

`/results` — the results of resolution are stored here