Universität Karlsruhe (TH)
Research University · founded 1825

Carnegie Mellon

Discriminative Word Alignment Models

Diplomarbeit
Interactive Systems Laboratories Prof. Dr. Alex Waibel
University of Karlsruhe (TH)
Carnegie Mellon University Pittsburgh (PA)

by

Jan Niehues

Advisor:
Prof. Dr. Alex Waibel
Dr. Stephan Vogel

Tag der Anmeldung:	1. Juli 2007
Tag der Abgabe:	21. Dezember 2007

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Karlsruhe, den 21. Dezember 2007

Jan Niehues

Jan Niehues

Abstract

In this thesis a new discriminative word alignment approach using conditional random fields is presented. The models using this approach differ in the knowledge sources they use, the way they are trained and the use of the regularization method. The approach models directly the alignment matrix. So every possible alignment can be generated by the models. Furthermore because this approach is symmetric no additional heuristics to combine both directions are needed.

In contrast to the common generative approaches it is possible to include all available information into the models. So the models are able to profit from the different outputs of the GIZA++-Toolkit or can make use of Parts-of-Speech(POS)-tags.

Since the structure of CRFs that model the word alignment matrix can become complicated, the standard algorithms for CRFs cannot be applied. So the inference can only be done approximately and the algorithm for the inference and training have to be adapted. It is only possible to use them efficiently if the algorithms make use of the special characteristics of some features used in the word alignment.

Different methods to train these models have been developed. Besides from the standard approach to maximize the log-likelihood the CRFs can be trained towards approximations of word alignment metrics. Furthermore these methods enable the system to be trained more towards precision or recall.

Since these models learn supervised, they have a tendency towards overfitting especially if the number of features grows. Therefore, a method to regularize that is well suited for CRFs is presented.

At last, the word alignment quality of the presented models is tested as well as the translation quality of a system using this word alignment. It will be shown that the translation quality improves significantly compared to the GIZA++-Toolkit. Furthermore the word alignment quality is at least as good as the best discriminative approaches presented in recent years.

Zusammenfassung

In dieser Arbeit wird ein neuer Ansatz zum Erstellen eines “Word alignments” für parallele Texte vorgestellt. Der präsentierte Ansatz benutzt “Conditional Random Fields (CRFs)” um die “Alignment”-Matrix zu modellieren. Die präsentierten Modelle unterscheiden sich in den benutzten Wissensquellen, den Trainingsmethoden und dem Verhindern von “Overfitting”. Dieser Ansatz ermöglicht es mit den Modellen beliebige Alignments zu erstellen. Außerdem ist dieser Ansatz symmetrisch, so dass keine zusätzlichen Heuristiken zum Kombinieren der “Alignments” beider Richtungen benötigt werden.

Im Gegensatz zum Standardansatz, dem GIZA++-Toolkit, ist der präsentierte Ansatz diskriminativ. Dadurch können einfach zusätzliche Features den Modellen hinzugefügt werden. Diese können benutzt werden, um alle verfügbaren Wissensquellen für das “Word alignment” in das System einzubinden. So kann zum Beispiel die syntaktische Funktion der Wörter benutzt werden, um zu bestimmen, welche Wörter Übersetzungen voneinander sind.

Die Strukturen der benutzten CRFs sind relativ komplex. Deshalb kann die Inferenz nicht exakt berechnet werden und stattdessen muss ein Approximationsalgorithmus benutzt werden. Dazu wird der schon in anderen Anwendungen für CRFs erfolgreich benutzte “Belief Propagation”-Algorithmus angewendet . Dieser musste allerdings weiter an die Anwendung angepasst werden, um auch Abhängigkeiten zwischen vielen Variablen modellieren zu können. Dazu werden die Eigenschaften der benutzten Features ausgenutzt.

Des Weiteren werden verschiedene Möglichkeiten entwickelt, um die Modelle zu trainieren. Zum einen wird die Standardmethode, die logarithmische Wahrscheinlichkeit zu maximieren, den Modellen angepasst. Zum anderen werden Methoden entwickelt, um die Modelle direkt bezüglich verschiedener ”Word alignment”-Metriken zu trainieren. Da auch der F-Score benutzt wird, ermöglicht dieser Ansatz, Modelle zu entwickeln, die mehr oder weniger “Links” erstellen.

Dabei kann es zu ”Overfitting“ kommen, da diese Modelle mittels ”überwachtem Training“ lernen. Dies ist im Besonderen der Fall, wenn die Anzahl der benutzten Features groß ist. Um dies zu verhindern wird eine Methode, die bereits bei anderen CRFs erfolgreich angewandt wurde, für diesen Ansatz adaptiert.

Zum Abschluss werden die Modelle evaluiert. Dazu wird zum einen die ”Word alignment“-Qualität der verschiedenen Systeme gemessen. Zum anderen wird die Übersetzungsqualität von Systemen gemessen, die die erzeugten ”Word alignments” benutzen. Diesen Experimenten zeigen, dass die Übersetzungsqualität mittels des neuen Ansatzes signifikant verbessert werden kann.

Contents

1. Introduction

In a globalized world the communication between people in different countries and continents grows. In the economy as well as in politics and other areas more and more people need to talk to people speaking different languages or read information in other languages.
Furthermore, the Internet enables people to access more and more information from all over the world. To be able to get this information fast can decide upon success and failure and the information is not always available in a common spoken language like English.
The solution to publish all information in one language, like for example English, cannot be the best way since the language is often an important part of the cultural identification.
Consequently, the need of text and speech translation grows constantly in many different areas. For example:

1. In politics: All speeches and documents of the European Union have to be translated into all 23 official languages of the European Union. In 2007 the European Union has to pay 302 million euro for these translations.

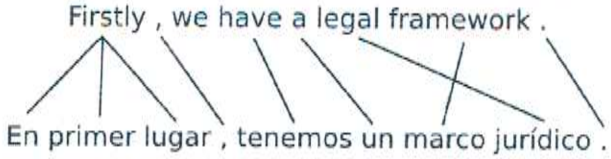
2. In economy: In the globalized world more and more companies cooperate with companies in other countries or sell their products abroad. Furthermore, international companies have many languages in their own ranks.

3. In tourism: In recent years the number of people spending their holidays in countries far away has increased. There they get into contact with people speaking different languages.

4. In humanitarian aid: In projects concerning humanitarian aid people often do not speak the same language as the one they want to help.

One way that could help to solve all these problems could be to improve the quality of machine translation and make machine translation available to more people. A very promising approach in this area is statistical machine translation.

Figure 1.1: Example of word alignment


Firstly , we have a legal framework .
En primer lugar , tenemos un marco jurídico .

1.1 Word alignment task

The statistical machine translation approach uses parallel data as one important knowledge source. In order to build good translation systems large corpora are needed. In current systems these corpora have millions of words in both languages. To use these corpora in the systems it has to be defined which parts of the source text are translated into which parts of the target text. To be able to produce such big corpora this is done on sentence level for most of them. This means, that for every source sentence it is known into which target sentence it is translated.

The problem of sentence aligned data is that it cannot be used in this format in the translation systems. Since the same sentence only occurs rarely more than one time, they cannot be used to build new translations. Instead, short parts of the text have to be used to build the translation for an unknown source sentence. Words or phrases of some words are the adequate length. To extract pairs of source and target phrases or words from the corpus an alignment between the source and target words has to be found.

An example for an alignment of an English-Spanish sentence pair can be seen in figure 1.1. English and Spanish words, which are translations of each other, are connected by a link. Of course, not every word is connected by a link as not every word has a corresponding word in the other language. In the example shown in Figure 1.1, "we" is not connected to any target word since this word has no direct translation in the Spanish sentence. But of course, another correct alignment of these sentences would be, if "we" is aligned to "tenemos", as the "-mos" indicates the first person plural. This small example already shows, that there is not always a unique correct alignment.

Furthermore, there are words in both languages, that are translated into several different words in the other language. Then the word is linked to all the words in the other language. For example, the English word "Firstly" is connected to the words "En", "primera" and "lugar". Sometimes a source and target phrase are translations of each other, but not a single word of the source side is a translation of a target word. In this case all words in the source phrase are linked to all words in the target phrase.

Another problem of the word alignment is, that the corresponding words in the source and target phrase are not always in the same order. In this case the links between words cross each other as you can see it in the word pairs "legal"-"jurdico" and "framework"-"marco" .

As mentioned before, this word alignment cannot be generated manually for corpora of several million words. Consequently, the word alignment has to be done automatically. Most of the approaches use statistical methods to determine the most probable alignment. Some will be described in section 1.3.

1.2 Goals

The phrases extracted from a parallel corpus using the word alignment are a very important knowledge source for a statistical machine translation system. Since the word alignment quality influences directly the quality of the phrases, a good word alignment is essential to generate high quality translations.

The most common word alignment approaches are the IBM- and HMM-models presented in [BPPM94] and [VoNT96]. They are implemented in the GIZA++-Toolkit which is used by many researchers. The problem of these approaches is that they are generative. Consequently adding additional information for the word alignment is quite complicated.

The goal of this work is to present new discriminative word alignment models. Discriminative models need to be trained on a hand-aligned development set, but if this data is available, these models have several advantages which enables them to generate better alignments.

One main advantage of discriminative models is that new information can be integrated easily into the model by using additional features. In European languages, for example, there are many cognates. These are words, which are translations of each other and which are written quite similar. For example the word "political" in English and "politica" in Spanish. Consequently, the similarity of two words should be used as an indication that these two words are translations of each other. This is nearly impossible in a generative approach but can be done quite easily in a discriminative framework. Consequently, the models presented should be able to use all available knowledge sources to generate the best possible alignment.

Furthermore the models should be symmetrical. This means that they will generate the same alignment for an English-Spanish sentence pair, whether the Spanish sentence is used as source or target sentence. In contrast, the GIZA++-Toolkit will generate different alignments. Then heuristics have to be used to combine both alignments.

After the discriminative word alignment framework is trained, the alignment is done independently for every sentence. This enables discriminative word alignment models to align big corpora in parallel. Consequently, more calculation capacity can be used for generating advanced word alignments fast.

In the end the main goal is to generate better word alignments by making use of the advantages mentioned above. This should not only be measured directly by using word alignment metrics like the alignment error rate (AER) but also by evaluating the translation quality of the SMT-systems using the improved word alignment. This is done since the main usage of the word alignment is to build SMT-systems.

1.3 Related work

In recent years several discriminative frameworks for word alignment have been presented to improve the word alignment quality[Moor05] [MoYB06] [TaLJK05] [LJTKJ06][BlCo06]. These frameworks can handle a large amount of features and make it easy to add new features. But the disadvantage is that they need to be trained in a supervised manner. Consequently, a small amount of hand-aligned data is needed.

All the presented approaches evaluate their systems on the French-English hand-aligned corpus which is part of the Canadian Hansard corpus. Since this contains only a devset of 37 sentences, the original test set was split into development data and test data. Two different versions were used. In one system, the first 100 sentences were used as development data and the rest as test data. In the other approach the data was split randomly into to virtual equal subsets. Taskar presented results on both versions in [TaLJK05] and the results were very similar.

One approach was presented by Moore in the publications [Moor05] and [MoYB06]. In this approach, a weighted linear combination of features is used to evaluate the alignment. The weights were trained on a hand-aligned set using a modified version of the average perceptron learning. To find the best alignment a beam search is applied, since not all possible alignments can be evaluated. The framework uses a two stage approach. In the first stage, each word-to-word link is considered separately and the link probability is modeled using the log-likelihood-ratio statistic. In the second stage, only links between disjoint clusters of source and target words were considered. To estimate these probabilities, the links between the clusters generated by the first stage were used.

In addition to the features used before, different features to model first order dependencies were used: for example the number of jumps and the length of the jumps. Furthermore, the number of unaligned words was used as a feature.

In an improved version of the framework presented in [MoYB06], it can also be trained using the support vector machine method. The best result was obtained by using also the IBM Model 4 links and the alignment probabilities obtained from a pair of HMM alignment models as a feature. Then the alignment was evaluated on the data version using 2 equal subset for development and test data and an alignment error rate (AER) of 3.7% on the test set was reached.

Another discriminative approach was presented by Taskar et al. in [TaLJK05]. Here, the word alignment problem was modeled as a maximum weighted matching problem. This problem can then be solved using a linear program (LP) formulation. The framework can be trained using two different methods. Either it can be trained using also the average perceptron method or a large margin method can be used. The best results were obtained by using the second method. During the training the error was measured using the hamming distance between the solution and the reference and the optimal solution was found using the extra-gradient method.

This approach has got two main drawbacks. First, the framework restricts the alignments to have a fertility of at most one. And second, it was not able to model first order dependencies. To overcome these problems, in a second version of the method presented in [LJTKJ06], the LP formulation was changed in a way that first oder dependencies can be modeled and several links per word are allowed. Using this improved framework and optimizing it on the first 100 sentences of the original test set an alignment error rate of 3.8% on the other part of the test set was obtained.

A third approach for word alignment was presented by Blunsom and Cohn in [BlCo06]. They used linear-chain conditional random fields (CRFs) to get a word alignment. Therefore, they trained a CRF for every direction and combined both alignments using the refined heuristic presented in [OcNe03] for IBM-alignments. This has to be done, since linear-chain CRFs are not able to model one-to-many alignments. The advantage of linear-chain CRFs is, that it is possible to find the optimal solution using dynamic programming. The CRFs were trained with a maximum a posteriori optimization using a Gaussian prior.

In the experiments different features were used to model the lexical probabilities, the fertilities and the first order dependencies. The best result on the English-French data they achieved was an alignment error rate of 5.29% when optimizing the system on the first 100 sentences like it was done by Taskar.

1.4 Structure of the work

In chapter 2 the basics for the thesis will be explained. First, the word alignment problem will be defined and a basic statistical machine translation(SMT)-system will be explained. Then, conditional random fields will be presented, as this is the discriminative model used for the word alignment in this thesis. Also the inference algorithm and regularization techniques are described. In the end, the word alignment metrics used in this work are presented.
In the next chapter, the models used for the word alignment will be presented in detail. Therefore, first the structure of the conditional random fields will be described in detail. Afterwards, the different features used in the experiments will be introduced.
In Chapter 4 the inference algorithm used in the models will be described. First, the inference problem is presented. Since the structure of the conditional random field can lead to complex calculations, some special techniques had to be used to make the inference efficient. They are presented in the rest of the chapter.
The fifth chapter is intended to give the details about the training of the new word alignment framework. In that chapter three different methods for the training process will be described.
Afterwards in section 6 a method to prevent overfitting in the discriminative word alignment framework will be presented. The described method is especially well suited for conditional random fields.
In last two chapters the results of the discriminative word alignment are presented. First, in chapter 7, the word alignment quality is evaluated and compared to other systems. Therefore, first the data used in the experiments is presented as well as the default configuration. Then the results using different knowledge sources are presented and the influence of the different features on the word alignment quality is evaluated. Afterwards, the models using the regularization techniques are investigated and a comparison to other approaches described in the literature is made.
In the second part of the results, the influence on the translation quality is evaluated. Therefore, different systems using the word alignment have been build. Especially, the different training methods are evaluated. In addition, the influence of the new word alignment models for a word reordering model is investigated.
Finally, in chapter 9, some conclusions and possible extensions in future work are discussed.

2. Basics

In this chapter the basics of the work will be presented. The next section concentrates on the word alignment. It will give the definition of a word alignment and describes the most common approach. In section 2.2, the basics of statistical machine translation (SMT) will be explained. SMT systems need a word-aligned corpus and it is the application of word alignment this work will concentrated on.
In section 2.3 conditional random fields (CRFs) will be introduced briefly. CRFs are the theoretical model used for the word alignment approach presented in this work. Afterwards, the belief propagation algorithm will be described since it is used for the inference in the CRFs.
Then in section 2.5 the basics of regularization techniques for CRFs will be described. In detail, the technique of logarithmic opinion pools for CRFs (LOP-CRFs) will be shown since this approach is applied to the models presented in this work.
At the end of this chapter, different word alignment metrics will be presented. These will be used later on to evaluate the word alignment approaches. Advantages and disadvantages of these different metrics will be discussed.

2.1 Word alignment

As the input the word alignment task gets a source sentence $f = f_1, \ldots, f_J$ and a target sentence $e = e_1, \ldots, e_I$. It is then the task to find the set of links (j, i) indicating that f_j is the translation of e_i. There are different simplifications of this task. The strongest one allows only $1:1$ alignments. This means, that each word e_i and f_j can only participate in at most one link. Then the alignment can be expressed as a injective function $a : J \rightarrow I + 1$ where $a_j = i$ means that the word f_j is aligned to e_i and $a_j = 0$ that the word f_j is not aligned. This simplification can be softened by allowing also $n:1$ alignments. This type of alignment can also be expressed as a function a, but it is no longer injective.
The word alignment was introduced by Brown et al. in [BPPM94] as a hidden variable in the translation process. They introduced different generative word alignment models IBM1,...,IBM5. These models are still the most common ones in the MT community. They are implemented in the GIZA++-Toolkit together with the HMM alignment introduced in [VoNT96]. These models have the advantage that

they are well suited for the noisy-channel translation systems and that they can be trained unsupervised. But they have also the disadvantage that it is difficult to add additional features like word similarity or syntactic information. Furthermore they need a lot of data to train and consequently large resources to process them.
The IBM-models are not symmetric since they can generate $n : 1$ alignments but no $1 : n$ alignments. So different alignments will be generated for a parallel corpus depending on which side is selected as source language. For SMT, it has been shown, that the performance can be improved, if a combination of both directions is used. A simple approach would be to use either the intersection or the union of both alignments. But by using different heuristics the results can be improved. These heuristics take all the links in the intersection and decide for all links that are in the union but not in the intersection whether to insert them into the alignment or not. Och and Ney proposed the "refined" heuristic to combine both alignments [OcNe03]. In [KoOM03] Koehn tried additional methods and evaluated their impact on the translation quality. They are all implemented in the Pharaoh-Toolkit.

2.2 Statistical machine translation (SMT)

Statistical machine translation (SMT) is one of the most promising approaches towards machine translation. Instead of manual created translation rules, the translations are learned from large bilingual corpora. This approach was presented the first time by Brown et al. in [BPPM94]. In recent years several researchers have improved this system.[WaWa98] [OcNe00] [YaKn01].
The SMT systems are based on the Bayes rule. Given a source sentence f_1^J the best translation e_1^I satisfies the following equation:

$$e_1^I = argmax_e p(e_1^I|f_1^J) = argmax_e p(f_1^J|e_1^I) * p(e_1^I) \tag{2.1}$$

This equation shows the three main problems of SMT. First, a good translation model to estimate the probability $p(f_1^I|e_1^J)$ is needed. The second problem is to model the probability $p(e_1^J)$ by the language model. At last, the argmax has to be found by the decoder.

2.2.1 Translation model

The translation model calculates the probability that the source sentence f_1^J is the translation of the target sentence e_1^I using information of large bilingual corpora. In the beginning of SMT this was done by using the word-to-word translation probabilities. To estimate this probabilities the mapping between the source and target word has to be known. This is defined by the word alignment. To be able to handle local renderings and encode context information, nowadays phrase-to-phrase probabilities are used to model the translation probability.
In the training of the translation model the word alignment has to be generated automatically, since the parallel corpora often are only aligned at sentence level. Then the phrase pairs that are consistent with the word alignment, are extracted. After the phrase pairs are extracted from the corpus, statistics about them were calculated.

2.2.2 Language model

The language model determines the probability $p(e_1^J)$ for a target sentence. The most common way to do this is to use n-grams. Then the probability for a target sentence is the product of the probabilities of all words given the $n-1$ previous words. These probabilities can be estimated from a large target language corpus.

2.2.3 Decoder

The decoder uses the information from the translation and language model to find a good or the best translation. The main problem is that the number of possible target sentences is very large. Consequently, not all possible translations can be evaluated. Instead, a beam search is performed, which only expands the best hypothesis.

2.3 Conditional random fields (CRFs)

Conditional random fields(CRFs) are an undirected graphical model introduced by Lafferty et al. in [LaMP01]. CRFs are a discriminative model that describes the conditional probability distribution $p(y|x)$ over some relational data, where x is a set of random variables that represent the observation and y is a set of random variables that represent the entities that should be predicted.

The relational data, that should be modeled, is characterized by two aspects. First, there is some statistical decency between the different entities in the model. And secondly, each entity has got a set of local features that express information for the classification. For example in the Parts-of-Speech(POS)-Tagging, there are local features that help to determine the POS-tag like the suffix or the capitalization of the word. Furthermore the POS-tag depends on the POS-tags of the neighboring words.

In contrast to generative graphical models, like Hidden Markov Models (HMMs), CRFs are discriminative graphical models. This means, that they do not model the joint probability $p(x, y)$ but the conditional probability $p(y|x)$. By doing this the distribution $p(x)$, which can have complex dependencies, is not needed. Furthermore, since no independence assumption has to be made, new features can be incorporated easily. In generative models, either the model has to be improved to model the dependencies between the different features or a simplifying independence assumption has to be made.

Another advantage of the CRFs compared to discriminative models like maximum entropy Markov models (MEMMs) is that there is no label bias problem. The reason for this is that the CRFs do not use a per-state exponential model, but have got a exponential model for the joint probabilities of all states.

Definition: Let X be a set of random variables and Y a set of target random variables. Furthermore, G is a graph on $V = X \cup Y$ specifying the dependencies between the random variables and C a set of cliques in the graph. In addition, for every clique a clique potential $\Phi_c(V_c)$, where V_c is the set of nodes in the clique, is defined. This is a log-linear combination of some features and it can be written as:

$$\Phi_c(V_c) = exp(\Theta * F_c(V_c)) = exp(\sum_k \theta_k * f_k(V_c)) \tag{2.2}$$

for a set of weights Θ and feature values $F_c(V_c)$.
Then the conditional random fields defines a conditional distribution $p(y|x)$:

$$p(y|x) = \frac{1}{Z(x)} \prod_{c \in C} \Phi_c(V_c) \tag{2.3}$$

$$= \frac{1}{Z(x)} \prod_{c \in C} exp(\Theta * F_c(V_c)) \tag{2.4}$$

where $Z(x)$ is the normalization factor defined as:

$$Z(x) = \sum_{y'} \prod_{c \in C} \Phi_c(V_c) \tag{2.5}$$

2.3.1 Representation

A CRF cannot only be described by the used graph. Sometimes it is preferable to represent the CRF by a factored graph. A factored graph is a bipartite graph consisting of two types of nodes. On the one hand there are the hidden nodes. These represent the variables for which the value should be inferred. On the other hand there are the factored nodes. These represent the cliques in the conditional random field. A factored node is connected to all hidden nodes that are in the clique represented by this factored node. This two representation are equal and can be transformed into each other.

2.3.2 Linear-chain CRFs

Although conditional random fields are defined on general graphs, in most applications only linear-chain CRFs are used. If the input and output data is sequential data, like it is for example in the POS-Tagging, shallow parsing and Named Entity Recognition [ShPe03] [McLi03], the graph has got a linear-chain structure. Then the cliques only consist of maximal two following output random variables and the input variable belonging to the first output variable.
The main advantage of linear-chain CRFs is that the inference can be done exactly by making use of dynamic programming. The training of linear-chain CRFs is done most of the time by maximizing the log-likelihood. There, the forward-backward algorithm can be used to limit the calculation costs.

2.3.3 General CRFs

Although they are more complex and the inference cannot be done exact, general CRFs have been used successful. For example, Taskar et al. used conditional random fields to label web pages [BTKo02].
The main problem is that the viterbi algorithm can no longer be used for the inference. Consequently, to calculate the inference in reasonable time, algorithms that only approximate the solution are used. Especially, since the inference has to be used in every training iterations a very fast method is needed. An algorithm that is often used is the belief propagation algorithm which will be described in detail in section 2.4.

2.4 Belief Propagation

The inference in the models described in this thesis is done using the belief propagation algorithm. The belief propagation algorithm is a message passing algorithm introduced by Pearl in [Pear88]. It is only exact for single connected graphs and even the convergence can only be shown for these graphs. But Yedida et al. showed in [YeFW03] that it has also a good performance on general graphs. Even if there are many loops in the graph, the algorithm will often lead to good results. The belief propagation will be described on factored graphs as it will be needed for the models presented in this thesis. This version of the algorithm can also be found in [LRHB06].

2.4.1 Motivation

The belief propagation algorithm infers the most probable values of the hidden nodes in an graphical model. Therefore, messages were passed between the nodes of the graph. Since the factored graph is a bipartite graph with two different types of nodes, there can only be messages from hidden nodes to factored nodes and messages in the inverse directions. The messages represent the local information of the sender about the receiver. Therefore, the other incoming messages and the local information are combined. This is an exact method for single connected graphs, but for other graphs it often gives a good approximation.

In the end the belief of one hidden node can be calculated from all incoming messages.

2.4.2 Algorithm

In single connected graphs the message calculation can start from a node, which has only one neighbor. Since the message is only calculated from all incoming messages except the inverse message, for this node the outgoing message can be calculated. Then the algorithm can continue to calculate always the messages, where all needed messages exist. For loopy graphs this is not possible. Consequently, the message needs to be initialized. In general, this is done by assigning the same value to every message to represent a uniform prior.

After having an initial value for every message, all messages can be calculated. Then in every iteration, first the messages $n_{(j.i)\to c}$ from the hidden node (j, i) to the factored node c is calculated. Afterwards the reverse messages $m_{c\to(j,i)}$ is calculated. The exact calculation of the messages is described in the next section. This is done for a fixed number of iterations or since the message values convergence. Afterwards the values of the hidden variables can be inferred from the incoming messages. This is described in detail in section 2.4.4.

2.4.3 Messages

First, there are the messages $n_{(j.i)\to C}$ send from a hidden node to the factored nodes. The message is represented by a vector of the size of possible values of the hidden node (j, i). The values are computed by the product of all incoming messages except the one coming from the factored node C.

$$n_{(j,i)\to c}(v) = \prod_{c' \in N(j,i)\setminus\{c\}} m_{c'\to(j,i)}(v) \tag{2.6}$$

where $N(j,i)$ is the set of neighboring factored nodes of (j,i).
On the other hand there are the messages $m_{c\to(j,i)}$ send from a factored node to a hidden node. These are also represented by a vector of the size of the possible values of the hidden node. Here, the computation of the value is defined as following:

$$m_{c\to(j,i)}(v) = \sum_{V'_c\in\mathcal{V}_c/v} \Phi_c(V'_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}((V'_c)_{(j,i)'}) \tag{2.7}$$

where $\mathcal{V}_c$ is the cross product space of the value space of all hidden nodes connected to this factored node. $\mathcal{V}_c/v$ are all possible values in $\mathcal{V}_c$, where the hidden variable (j,i) has got the value v. $N(c)$ are all hidden nodes connected to the factored node c and $(V'_c)_{(j,i)'}$ is the value of $(j,i)'$ in V'_c, the set of all values of hidden nodes in $N(c)$. Φ_c is the potential function of the clique represented by the factored node c.

2.4.4 Inference

After running the algorithm for several iterations the values of the hidden variables can be inferred from the messages send to the hidden variables. Therefore, the belief value of an assignment of a hidden node can be calculated as:

$$b_{(j,i)}(v) = \prod_{c\in N(j,i)} m_{c\to(j,i)}(v) \tag{2.8}$$

The belief value can then be interpreted as the marginal posterior probability. Then the most probable value as well as the expectation value of the hidden node can be easily calculated.

2.5 Regularization

The research on CRFs in different applications has shown, that CRFs have a tendency towards overfitting. To avoid this, different regularization techniques have been applied to CRFs. The most common one is to use a prior distribution for the parameters. This distribution encodes prior knowledge about the parameters into a probability distribution. But since this distribution can have a complex structure in most cases a simplification is made and a Gaussian prior with a zero mean and a constant variance is used.
One problem of this approach is, that an additional hyper-parameter search has to be applied, if the prior knowledge should be represented well. The distribution as well as good means and variances have to be found. In addition, some authors tried to use different distribution parameters for different features.
In the next section another approach to regularize CRFs, the logarithmic opinion pool (LOP) of CRFs(LOP-CRFs), will be presented. It was introduced by Smith et al. in [SmCO05] and [SmOs05].

2.5.1 Logarithmic opinion pools for CRFs (LOP-CRFs)

The LOP-CRFs model the distribution as a weighted product of the different individual expert CRF distributions. The main advantage of this approach is, that it is parameter-free and consequently no hyper-parameter search is needed. Furthermore, because CRFs are log-linear models they are well suited for the LOP framework. So LOP-CRFs are easy to implement and not computationally intensive.

Given the distribution of the different experts $p_\alpha(y|x)$ the distribution $p_{LOP}(y|x)$ of the LOP-CRFs is defined as:

$$p_{LOP}(y|x) = \frac{1}{Z_{LOP}(x)} \prod_\alpha \left(p_\alpha \left(y|x\right)\right)^{\omega_\alpha} \tag{2.9}$$

where the weights ω_α meet the requirement $\omega_\alpha \geq 0$ and $\sum_\alpha \omega_\alpha = 1$. Furthermore the normalization factor $Z_{LOP}(x)$ is defined by:

$$Z_{LOP}(x) = \sum_y \prod_\alpha \left(p_\alpha \left(y|x\right)\right)^{\omega_\alpha} \tag{2.10}$$

An interesting point about which experts should be used in a LOP can be made by looking at the KL divergence between the "true" conditional distribution $q(y|x)$ and the distribution modeled by the LOP-CRFs. In [Hesk98] it has been shown that the KL divergence can be written as:

$$KL(q, p_{LOP}) = \sum_\alpha \omega_\alpha KL(q, p_\alpha) - \sum_\alpha \omega_\alpha KL(p_{LOP}, p_\alpha) = E - A \tag{2.11}$$

This equation shows that a trade-off has to be made to minimize the error of the LOP distribution. The first term is minimized if the experts succeed in modelling the "true" conditional distribution very well. Consequently, all experts should model the distribution well. On the other side, to maximize the second term, the experts should be as different as possible. So experts should be used that are good themselves, but as different to each other as possible. For example, experts could be used to model different aspects of the problem. Then the weights can be used to describe the confidence of the different experts.

From equation 2.9 it can be derived that a LOP-CRF is also a simple CRF. This will be shown in detail in chapter 6. Because of this, the decoding in a LOP-CRFs is identical to the one in a normal CRFs and can be done efficiently. The optimization is also very similar. The only difference is that not the parameters of the CRFs are changed, but the weights of the experts. So the same optimization techniques can be applied as in normal CRFs.

2.6 Word alignment metrics

If working with different word alignment methods an evaluation metric is needed. This metric should be automatic to evaluate many different methods in a short time. Therefore, in recent years, different methods have been proposed, but nearly all of them compare the generated alignment to a gold standard alignment. Therefore, two different annotation schemes are used. Either the gold standard consists out of one set of links or the gold standard allows sure links S and possible links P. Then these possible links are used to model ambiguous alignments and free translations.

If the word alignment is used for machine translation the word alignment metric should predict the MT quality. Therefore, a good correlation between the word alignment metric and the translation quality is needed. But, for example in [AyDo06], it is shown that the correlation between the alignment quality and the translation quality is not easy. For example, phrase extraction methods and weighting functions perform differently, if they are used with different word alignments.

The most common metric for the word alignment quality is the alignment error rate

(AER) which was introduced in [OcNe03]. The definition is derived from the F-measure and can be calculated for a hand-aligned test corpus. It allows sure and possible links and is defined as:

$$AER(S,P;A) = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|} \tag{2.12}$$

where A is the generated alignment, S are the sure links and P are the possible links.
One problem of the AER is that it does not penalize an unbalanced precision and recall ratio. In [FrMa07] it is demonstrated that the correlation between the AER and the translation quality is not good. For example, the AER performance of a word alignment method can be improved by only selecting less links, but this would not improve the MT quality.
Another word alignment metric, the F-measure, was proposed by Fraser and Marcu in [FrMa07]. The F-measure is already successfully used in other areas. They extended the definition of the F-measure in a way that is able to handle sure and possible links. Then it is defined as:

$$F - measure(S,P;A,\alpha) = \frac{1}{\frac{\alpha}{Precision(P;A)} + \frac{1-\alpha}{Recall(S;A)}} \tag{2.13}$$

with:

$$Recall(S;A) = \frac{|S \cap A|}{|S|} \tag{2.14}$$

$$Precision(P;A) = \frac{|P \cap A|}{|A|} \tag{2.15}$$

By using the default value of $\alpha = 0.5$ they showed that the correlation between the F-measure and the BLEU score, as metric for the MT quality, is better than the one with the AER, but still there is still a lot to improve. But for French-English and Arabic-English, they got good correlations when they tried different values for α. They got the best results for values of α smaller than 0.5. This suggests that at least for their task the recall is more important than the precision.
In [AyDo06] Adan and Dorr introduced "consistent Phrase Error Rate (CPER)“. This metric does not look at the links itself but at the phrases that are consistent with the word alignment. So the metric does depend on the phrase extraction method and the results are only useful for systems using the same phrase extraction method. This metric does only accept one type of links in the gold standard alignment. Then, the phrase recall and phrase precision are defined analog:

$$PhraseRecall(G,A) = \frac{|G \cap A|}{|G|} \tag{2.16}$$

$$PhrasePrecision(G,A) = \frac{|G \cap A|}{|A|} \tag{2.17}$$

And the CPER is defined as:

$$CPER(G,A) = 1 - \frac{2 * PhraseRecall(G,A) * PhrasePrecision(G,A)}{PhraseRecall(G,A) + PhrasePrecision(G,A)} \tag{2.18}$$

The authors indicate that the CPER is a better prediction for the translation quality than the AER and in contrast to Fraser et al. they get the best results with word alignment methods that are more precision-orientated.

As already mentioned, the task to predict the translation quality from the word alignment is complicated. Several researches have investigated this problem in recent years. In [ViPN06], for example, the alignment was adapted towards the translations models. For a phrase-based system, for example, they removed alignment links that have a big distance to all other alignment links, because this links prohibit to extract many phrases. It is shown on a German-English system, that this leads to a worse AER, but improves the translations. So the AER or the F-measure may be a good measure for the word alignment quality, but because of the inconsistency between the word alignment models and the models for the translation process, this does not have to lead to to a good translation quality. So they proposed to include the word alignment into the training procedure of the later models.

Another point of the correlation between word alignment metrics, in this example AER, and the translation quality is made by Wu and Wang in [WuWa07] by looking at in-domain testing data and out-of-domain testing data. They showed that they get the best results for AER by having an alignment which makes a good compromise between recall and precision. The best results for in-domain translations they get with an alignment that is more recall-oriented. In contrast, if they translate an out-of-domain test data, they get the best results if they use a precision-oriented alignment. One reason is, that more phrase are extracted with a precision-oriented alignment and so they have less unknown words.

3. The models

All models presented in this work use conditional random fields (CRFs) to model the alignment matrix directly. So they are able to find many-to-many alignments and there is no different treatment for source and target language. Consequently heuristics to combine both directions are no longer needed. In the next section the structure of the conditional random fields will be described.
The presented word alignment models differ in three main areas. First, the different models use different knowledge sources. These can be used by the features of the models, which could be easily included to the models, since CRFs are discriminative models. In section 3.2 the features used in the different models will be shown.
Furthermore, there are different methods to train the models. These are described in chapter 5. At last, in section 6 a regularization method is presented, which can also be used in the models. But not all models use this approach, since additional training data is needed.

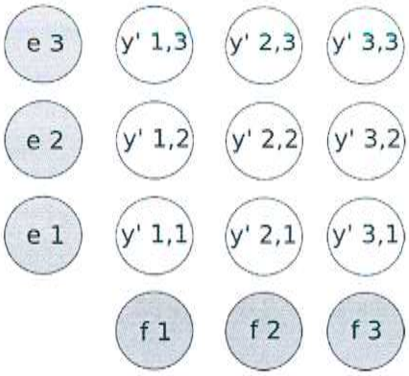
3.1 Structure

The models consist of a Graph G over the random variables X and Y', where X are the observed variables and Y' the labels. In this graph $X = X_f \cup X_e = \left\{x_j^f | 1 \leq j \leq J\right\} \cup \{x_i^e | 1 \leq i \leq I\}$ where J is the source sentence length and I is the target sentence length. The variable x_j^f corresponds to the source word f_j and the variables x_i^e represent the target words e_i. For every source and target word pair (f_j, e_i) there exists a hidden random variable y'_{ji} having the value 0 or 1 to indicate if there is a link between these words or not. The nodes of an example graph are shown in figure 3.1.

As described in chapter 2.3 the probability is defined by a product of potential functions of the cliques in the graph. If the set of cliques $C = \{c\}$ is given, the probability can be calculated:

$$p_\Theta(y|x) = \frac{1}{Z(x)} \prod_{c \in C} \Phi_c(V_c) \tag{3.1}$$

Figure 3.1: Model structure



where $Z(x) = \sum_Y \prod_{c \in C} \Phi_c(V_c)$ is the normalizing factor over all possible alignments, V_c the set of nodes in the clique c and Φ_c is a potential function.
The potential function Φ_c is defined in CRFs as following:

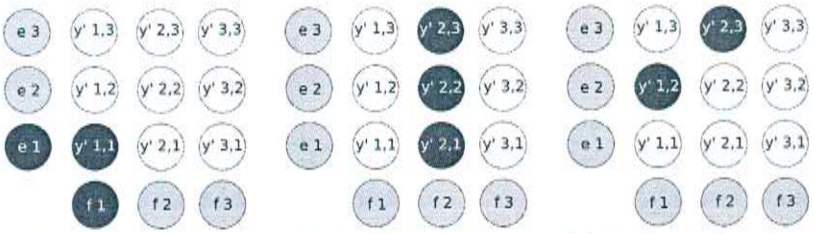
$$\Phi_c(V_c) = exp(\Theta * F_c(V_c)) = exp(\sum_k \theta_k * f_k(V_c)) \tag{3.2}$$

where Θ is the vector of the scaling factors and $F(V_c)$ the feature vector of this clique c. To complete the description of the structure the cliques used in the models will be described next.
First, there are the cliques consisting of a random variable x_i and x_j and the corresponding hidden variable y'_{ji}. The features in these cliques will be called local features in the next section, because their value is only influenced by the word pair and not by the interaction between different links. The features of this type of clique are described in detail in section 3.2.1 and an example is shown in figure 3.2(a).
The second type of cliques contains all hidden variables y'_{ji} belonging to one source or to one target word. The corresponding features are called fertility features because they look at all links of a word and can control the fertility of this word.
The next type of cliques models the first order dependencies. These cliques consist of two hidden variables $V_c = (y'_{ji}, y'_{j+s,i+t})$. The features in all cliques consisting of pairs $V_c = (y'_{ji}, y'_{j+s,i+t})$ will be called first order features for the directions (s, t), since they model first order dependencies between links which are connected in the direction (s, t). For example, for the majority of language pairs the most important first order features have the direction $(1, 1)$.
Finally, there are cliques for phrase features to model local dependencies between words. For example, the link between two articles can depend on the words that are following in the two sentences. To model these dependencies all links, that are important for the phrase pair, are part of the clique. These can be only the variables representing the links between words of both phrases or all links of whom one word is in the phrase pair.

3.1.1 Factored graph

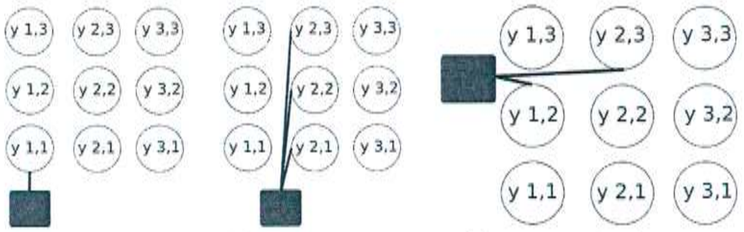
Since the graphs of the described models have many loops, the inference in these models cannot be done exactly. Instead, this is only done approximately using the belief propagation algorithm. This is still complicated, since the runtime is normally exponential in the size of the cliques. To reduce the calculation time

Figure 3.2: Different features



(a) Local features (b) Fertility features (c) First order features

Figure 3.3: Different features



(a) Local features (b) Fertility features (c) First order features

only special feature functions are used in large cliques. The modifications to the belief propagation algorithm are described in the next chapter. To be able to make these modifications it is better to perform the belief propagation not in the original graph but on the corresponding factored graph. The method to convert the graph is described in [YeFW03].
The resulting graph is bipartite and consists of two different kinds of nodes. First, there are the variable nodes. The variable nodes of the factored graph correspond to the former hidden random variable nodes. Furthermore, there are the factored nodes (rectangles in the image) corresponding to the cliques in the other graph. A factored node is connected to a variable node only if the variable was in the corresponding clique. The resulting graph $G_F = (V_F, E_F)$ is defined in equation 3.3 and an example is shown in figure 3.3.

$$\begin{aligned} V_F &= V_{HN} \cap V_{FN} \\ V_{HN} &= \{y_{ji} | 1 \leq j \leq J, 1 \leq i \leq I\} \\ V_{FN} &= \{x_c | c \in C\} \\ E_F &= \{(y_{ji}, x_c) | y_{ji} \in C\} \end{aligned} \tag{3.3}$$

The probability in 3.1 can then be written as follows:

$$p_\Theta(y|e,f) = \frac{1}{Z(e,f)} \prod_{x_c \in V_{FN}} \Phi_c(V_c) \tag{3.4}$$

where V_c is the set of variable nodes connected to the factored node c.

3.2 Features

As mentioned before there are four different kinds of features. First, the local features will be presented. Afterwards, in 3.2.2, the fertility features will be described. In the next sections the first order features and the phrase features will be presented. In the experiments using these models the features are often grouped into two parts. Then all features except the POS-based and high frequent words features will be called baseline features since they are used in most of the models. The other features will be referred to as additional features. Normally there are many more additional features than baseline features. For example, using high frequent word features for the 50 most frequent words leads to 2500 additional features.

3.2.1 Local features

Local features are those features that concentrate only on one pair of words and that do not depend on the neighborhood of the source and target word. Most of them model the probability that the source and target word are translated into each other. But there are also others like a feature for the difference in the relative sentence position. These model the fact, that words at the beginning of a source sentence are more likely translated into words at the beginning of the target sentence.

1. For each lexicon two features were added to the model. These features are:

   $$\begin{aligned} f_{lex0}(\{y_{ji}\}) &= (1 - p_{lex}(f_j, e_j)) * 1_{y_{ji}=0} \qquad (3.5)\\ f_{lex1}(\{y_{ji}\}) &= p_{lex}(f_j, e_i) * 1_{y_{ji}=1} \qquad (3.6)\end{aligned}$$

   The first feature should help to prevent words with low translation probability to be aligned to each other. The second feature is designed to align words with a very high translation probability. If this feature gets a positive scaling factor and two words have a higher lexical translation probability, they will be aligned more probably.

   In the standard setup the models use two lexica, the IBM lexica of both directions. But, of course, it is easy to add other lexica to the models and, for example, manual lexica could be used.

2. Furthermore there are features which use a normalized lexical probability. The lexical probability is normalized in two different ways, that either for one source or target word the lexical probabilities in one sentence sum up to one. This leads to the following definitions:

   $$\begin{aligned} f_{sourceNormLex0}(\{y_{ji}\}) &= (1 - p_{sourceNorm}(f_j, e_i)) * 1_{y_{ji}=0} \qquad (3.7)\\ f_{sourceNormLex1}(\{y_{ji}\}) &= p_{sourceNorm}(f_j, e_i) * 1_{y_{ji}=1} \qquad (3.8)\\ f_{targetNormLex0}(\{y_{ji}\}) &= (1 - p_{targetNorm}(f_j, e_i)) * 1_{y_{ji}=0} \qquad (3.9)\\ f_{targetNormLex1}(\{y_{ji}\}) &= p_{targetNorm}(f_j, e_i) * 1_{y_{ji}=1} \qquad (3.10)\\ p_{sourceNorm}(f_j, e_i) &= \frac{p_{lex}(f_j, e_i)}{\sum_{1 \le j \le J} p_{lex}(f_j, e_i)} \qquad (3.11)\\ p_{targetNorm}(f_j, e_i) &= \frac{p_{lex}(f_j, e_i)}{\sum_{1 \le i \le I} p_{lex}(f_j, e_i)} \qquad (3.12)\end{aligned}$$

   This helps to align words of which all lexical probabilities are quite small. It indicates if there are several good translations for one word or if there is only one.

3. The next local feature should help to align words that are in a similar position in the sentence. For example, a word in the beginning of the source sentence is more likely translated into one at the beginning of the target sentence than to one in the middle or even at the end. This is especially important if there are words that occur several times in one sentence, for example in the target sentence. Then the lexical features cannot help to decide for a source word to align this word to which one of equal target words. These features are defined as follows:

$$\begin{aligned} f_{distance0}(\{y_{ji}\}) &= \left(1 - \left|\frac{i}{I} - \frac{j}{J}\right|\right) * 1_{y_{ji}=0} && (3.13) \\ f_{distance1}(\{y_{ji}\}) &= \left|\frac{i}{I} - \frac{j}{J}\right| * 1_{y_{ji}=1} && (3.14) \end{aligned}$$

4. Another feature is the equality feature. This feature indicates if the source and target words are equal. This is useful for names and numbers which are often the same in both languages. The only problem is that, for example, commas are equal, but often the number of commas is not equal in the source and target text. Consequently, commas are not always aligned to another one, which could cause some problems. So this feature is only one if the source and target words are longer than 1 character each.

$$\begin{aligned} f_{euqal0}(\{y_{ji}\}) &= 1_{e_i \neq f_j \vee |e_i|=1 \vee |f_j|=1} * 1_{y_{ji}=0} && (3.15) \\ f_{equal1}(\{y_{ji}\}) &= 1_{e_i = f_j \wedge |e_i|>1 \wedge |f_j|>1} * 1_{y_{ji}=1} && (3.16) \end{aligned}$$

5. In addition, there is a similarity feature. This calculates the edit distance between the source and target word. To have again a value between 0 and 1 the edit distance is divided by the maximum of source and target word length. This feature is used because, for example, in European languages there are many similar words that are translated into each other like the English and Spanish words "politica" and "political". The values of these features are calculated as following:

$$\begin{aligned} f_{sim0}(\{y_{ji}\}) &= \left(1 - \frac{EditDis(e_i, f_j)}{max(lenght(e_i, f_j))}\right) * 1_{y_{ji}=0} && (3.17) \\ f_{sim1}(\{y_{ji}\}) &= \frac{EditDis(e_i, f_j)}{max(lenght(e_i, f_j))} * 1_{y_{ji}=1} && (3.18) \end{aligned}$$

6. The next type of features is the link feature. This feature can use the word alignment of other systems as additional knowledge source. Therefore, let a be the set of links generated by another word alignment system and $w_{(j,i)}$ be the weight assigned to the link (j, i). This weight can be used to encode the confidence in the links. Then the features are defined as following:

$$\begin{aligned} f_{link0}(\{y_{ji}\}) &= \left(1 - w_{(j,i)}\right) * 1_{(j,i) \in a} * 1_{y_{ji}=0} && (3.19) \\ f_{link1}(\{y_{ji}\}) &= w_{(j,i)} * 1_{(j,i) \in a} * 1_{y_{ji}=1} && (3.20) \end{aligned}$$

7. Furthermore there are Parts-of-speech(POS) features. There is one POS-feature for each possible combination of a source and target POS-tag. Consequently, during the optimization for every combination, it can be decided if

the probability of a link for this combination of tags should be increased or decreased by changing the scaling factor. This leads to the following definition for the features:

$$f_{pos(a,b)}(\{y_{ji}\}) = 1_{POS(e_i)=a} * 1_{POS(f_j)=b} * 1_{y_{ji}=1} \quad (3.21)$$

where $POS(w)$ is the POS-tag for the word w.

8. Similar to the POS features there are features for the high frequent words. For every combination of the n most frequent words of the source side and the target side, there is one feature, which is one only if the source and target word of the link equals that combination. This leads to the similar term:

$$f_{hfreq(a,b)}(\{y_{ji}\}) = 1_{e_i=a} * 1_{f_j=b} * 1_{y_{ji}=1} \quad (3.22)$$

3.2.2 Fertility features

The next group of features are the fertility features. They model the probability of a word to be translated into the empty word, into one, two and more words. The corresponding function node for one source word in the factored graph is connected to all I variable nodes representing links to this source word. Similarly, the function node for a target word is connected to all J nodes describing a link of this target word to a source word. These factored nodes use the following features:

1. For every fertility up to a given N there is one feature $f_{fertSource(n)}(\{y_{ji}\}_{1\leq i\leq I})$ indicating if this source word has a fertility of n and the same features for target words. The default value of N is 3.

$$f_{fertSource(n)}(\{y_{ji}\}_{1\leq i\leq I}) = 1_{\sum_i y_{ji}=n} \quad (3.23)$$
$$f_{fertTarget(n)}(\{y_{ji}\}_{1\leq j\leq J}) = 1_{\sum_j y_{ji}=n} \quad (3.24)$$

2. The last feature is only calculated up to a constant N to reduce the complexity of the inference for long sentences. But there is another feature indicating if the word has a fertility greater than N. This is defined as:

$$f_{fertSource(N)}(\{y_{ji}\}_{1\leq i\leq I}) = 1_{\sum_i y_{ji}\geq N} \quad (3.25)$$
$$f_{fertTarget(N)}(\{y_{ji}\}_{1\leq j\leq J}) = 1_{\sum_j y_{ji}\geq N} \quad (3.26)$$

3. The features described above are equal for every source and every target word. To model the individual fertility probabilities the values estimated by the GIZA++-Toolkit can be used as a feature. Again the complexity is reduced by using an approximation for large fertilities. Let $P(n|e)$ be the probability that the word e is translated into n words. Then the GIZA-fertility feature is calculated in the following way:

$$f_{GIZAFertSource}(\{y_{ji}\}_{1\leq i\leq I}) = P_{source}(\sum_i y_{ji}|e_i) \quad (3.27)$$
$$f_{GIZAFertTarget}(\{y_{ji}\}_{1\leq j\leq J}) = P_{target}(\sum_j y_{ji}|f_j) \quad (3.28)$$

3.2.3 First order features

The next group of features are the first order features, which model the first order dependencies between the links. They are grouped into different directions. A factored node for the direction (s, t) is connected to the variable node y_{ji} and $y_{(j+s)(i+t)}$. Â Small positive and negative numbers are used to cover the different directions. For example, the most common direction is $(1, 1)$ which describes the situation that if the words at position j and i are aligned, also the immediate successor words in both sentences are aligned. In the Spanish-English task the direction $(1, -1)$ is also important since noun and adjective are often in different order in the languages. The following features will be calculated for every direction.

1. The first feature is 1 if both nodes connected to the function node represent no link. This leads to the following definition:

$$f_{FirstOrder0}(\{y_{ji}, y_{(j+s)(i+t)}\}) \;=\; 1_{y_{ji}=0} * 1_{y_{(j+s)(i+t)}=0} \qquad (3.29)$$

2. The next feature is 1 if exactly one node represents a link

$$f_{FirstOrder1}(\{y_{ji}, y_{(j+s)(i+t)}\}) = 1_{y_{ji}=1 \wedge y_{(j+s)(i+t)}=0} + 1_{y_{ji}=0 \wedge y_{(j+s)(i+t)}=1} \qquad (3.30)$$

3. Then there is one feature indicating if both nodes correspond to a link

$$f_{FirstOrder2}(\{y_{ji}, y_{(j+s)(i+t)}\}) \;=\; 1_{y_{ji}=1} * 1_{y_{(j+s)(i+t)}=1} \qquad (3.31)$$

4. In addition there are two types of POS-based features. One depends on the POS-tags of the words corresponding to the first node y_{ji}, the other depends on the POS-tags corresponding to the second node $y_{(j+s)(i+t)}$. There is one feature for every possible combination of source and target POS-tag like it was done for the POS-based local features. The value of the features can be calculated as follows:

$$f_{POSStart(a,b)}(V_c) \;=\; 1_{y_{ji}+y_{(j+s)(i+t)}=2 \wedge POS(e_i)=a \wedge POS(f_j)=b} \qquad (3.32)$$

$$f_{POSEnd(a,b)}(V_c) \;=\; 1_{y_{ji}+y_{(j+s)(i+t)}=2 \wedge POS(e_{i+t})=a \wedge POS(f_{j+s})=b} \qquad (3.33)$$

   with $V_c = \{y_{ji}, y_{(j+s)(i+t)}\}$.

3.2.4 Phrase features

The last type of features are the phrase features. These features are designed to model the context dependencies of a link. For example, the word "the" on the English side is more probable aligned to "le" or "la" depending on the following word on the English side. Therefore, in a first step, the whole corpus is aligned with a model having all the other features. From this aligned corpus phrase pairs are extracted and statistics about the phrase pairs will be used in a second alignment step.

There are two different ways to build these phrase pairs. The one method builds a phrase table and the other a group table. The difference between both methods will be described later, but in both cases there exists a source phrase $f_{j_1}^{j_2}$, a target phrase $e_{i_1}^{i_2}$, the links a between the words in the source and target phrase and several

values v_k that are assigned to this phrase pair. In the discriminative word alignment models there are two different ways, for how the features are calculated.
First, in the "normal mode" only the variables that represented the links in the alignment a of the phrase pair are considered. The feature is then defined as:

$$f_{phrase1}(\{y_{ji}\}_{(i-i_1,j-j_1)\in a}) = v_k * \prod_{(i',j')\in a} 1_{(i_1+i',j_1+j')=1} \tag{3.34}$$

Let (i_1, j_1) be the starting sentence indices of the phrase pair. Then for every link in the alignment of the phrase pair, $(i', j') \in a$, the variable $y_{(i_1+i')(j_1+j')}$ is considered. If all these variables are one, the feature has the value v_k, otherwise the feature value is 0. So in this version it does not matter if the source or target words are linked to other words, too. But this should not be a problem since the fertility features are designed to avoid a word linked with too many other words. The main advantage of this method is that only a few variables are considered so that the calculation time is shorter.
The second mode is called "exact mode". In this mode all variables representing a link of a word in the source or target phrase are considered. The feature is defined as:

$$f_{extPhrase}(V_{extPhrase}) = v_k * \prod_{(j',i')\in a} 1_{(i_1+i',j_1+j')=1} * \prod_{(j',i')\notin a} 1_{(i_1+i',j_1+j')=0} \tag{3.35}$$

with $V_{extPhrase} = \{y_{ji} | i_1 \leq i \leq i_2 \vee j_1 \leq j \leq j_2\}$ where (i_1, j_1) are the staring indices of the phrase pair and (i_2, j_2) are the indices of the end of the phrase pair. In this mode all variables corresponding to the links of the phrase pair have to be one, too, which is expressed by the first product. But the words of the phrase pair are not allowed to participate in a link which is not part of the phrase alignment a. This restriction is described by the second product. So the feature will prevent alignments between words inside and outside the phrase. The main problem is that the calculation time does increase.
The phrase and group tables are extracted in the following way:

1. The phrases are extracted from the corpus like it is done in the phrase extraction for the translation model. For all source phrases longer than one word and up to a given maximal length all target phrases that do agree with the alignment are extracted. A source and target phrase pair agrees with the alignment if no word from inside a phrase pair is aligned to a word outside the phrase pair. In the experiments a maximal length of 4 is used.
   The following types of values were used to evaluate a phrase:
   (a) The first value is a relative frequency value. It is the number of times the source and target phrase occur in the text with this alignment divided by the number of times the source and target phrase co-occur.
   (b) Then there are features indicating if the source and target length is the same, the source side is longer or there are more words in the target phrase.
   (c) At last, there are two features describing how many words of the phrase are linked. The first is defined by the number of source words that are linked with target words divided by the number of source words. The second feature is calculated in the same way by using the target words.

2. The group table is build by extracting groups of source and target words from the corpus. Words will be in the same group if all source words are aligned to all target words. In this case for every aligned sentence pair there exists only one way to split the words into groups. All groups that are non-continuous on the source or target side are ignored. The set of links, a, consists of all possible combinations of source and target indices since all source words are aligned to all target words.

   The feature values used for the groups are calculated in the following way:

   (a) First, there is again a relative frequency feature. This is calculated by the number of times all the source words are aligned to all target words divided by the co-occurrence of the source and target words.

   (b) Then there are indicator features describing whether this group represents a $1:1$, $1:n$, $n:1$, $n:m$ link.

4. Inference

After defining the models in the last section, this section will concentrate on the inference task. First, the problem of inference will be described in detail in the next section. In section 4.2 it will be explained how the belief propagation algorithm described in section 2.4 can be used to approximate a solution for this task. Afterwards the calculation of the messages used during the belief propagation algorithm will be described in detail. In the end the creation of an alignment from the result of the algorithm will be explained.

4.1 Problem description

Using the models presented in the last chapter the word alignment problem equals the inference problem in the described conditional random fields. Given a source and target sentence the most probable assignment of the hidden variables according to the posterior distribution $p(y|x)$ has to be found. As mentioned in section 2.3 this can only be done exactly for linear-chain CRFs using a variation of the Viterbi algorithm. But since the structure of the presented models is more complex, the exact inference is completely intractable. Instead, in the models the belief propagation algorithm described in 2.4 is used as proposed by [TaAK02] and [LRHB06].

4.2 Algorithm

The inference is approximated using the belief propagation algorithm introduced by [YeFW03]. The algorithm is described in detail in 2.4. In the models it is used to find the most probable assignment of the hidden variables. Since the hidden variables can have two different values, 0 and 1, one message in the models always consists of two values. To avoid the numerical problem that these numbers get too high or too low, they are always normalized in a way that the sum of both values is one.

All messages were initialized with the values $(0.5, 0.5)$ except the messages send by the local feature nodes. As it will be shown in 4.3.2, these messages do not depend on any other message. So they can be calculated once before the inference and can already be used in the first iteration.

In one iteration first the messages from the hidden nodes to the factored nodes will be calculated and afterwards the messages in the other direction. By default, this will be done for 10 iterations because using more iterations seems not to improve the system performance.

4.3 Message calculation

Using the belief propagation algorithm the only remaining problem is to calculate the messages sent during the algorithm effectively. This cannot be done straight forward for all messages in these models, since the common approach has an exponential complexity in the clique size and the cliques in the propose models can be quite big. So it is not possible to present a common way to calculate the message values. But different ways will be shown for the different types of nodes in the factored graph. First, in part 4.3.1, the calculation of the first type of messages from a hidden node to a factored node will be described. In the remaining parts the other type of message, from a factored node to a hidden node, will be described. Since there are different factored nodes corresponding to the different types of cliques in the original graph, this will be done for every type of factored node separately.

4.3.1 Hidden nodes

As described earlier, the values of the messages send from a hidden node to a factored node is defined as:

$$n_{(j,i)\to c}(v) = \prod_{c' \in N(j,i)c} m_{c'\to(j,i)}(v) \tag{4.1}$$

For every message, sent from a hidden node, the calculation is linear in the number of connections. So for every hidden node this leads to a quadratic complexity in the number of connections. Consequently, the calculation for a hidden node can be done straight forward.

4.3.2 Local features

Since a factored node for the local features is only connected to one hidden node, the messages send from a local node to a hidden node are independent of the incoming messages as is shown in equation 4.2.

$$\begin{aligned} m_{c\to(j,i)}(v) &= \sum_{V_c/v} \Phi_c(V_c) \prod_{(j,i)' \in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\ &= \sum_{V_c/v} \Phi_c(V_c) * 1 \\ &= \Phi_c(v) \end{aligned} \tag{4.2}$$

So the message calculation can be done before the belief propagation algorithm starts.

4.3.3 Fertility features

The calculation for the fertility feature nodes is the most problematic one since this node can have many connections. One fertility node is either connected to all hidden nodes of one source or of one target word. Consequently, the calculation cannot be done straight forward, since it has to be summed up over all possible combination of incoming messages. This would lead to a sum of 2^I or 2^J parts, which could not be calculated in acceptable time for long sentences. But by using dynamic programming and only special features the messages can be calculated quite fast.

Therefore, the features used to model the fertility of a word have to be investigated in detail. For these features it is only important how many nodes are active and not which nodes are active. This leads to a dynamic programming approach, where only the probability of $1, 2, \dots, N$ nodes being active is calculated. This can be done in $O(N * I)$ or $O(N * J)$ for one connection using the recursion described below and so the calculation of all the messages of one node can be done in $O(N * I^2)$ or $O(N * J^2)$.

In the following definitions a source fertility node is used, but the target side can be done the same way by just switching j and i.

First, for every message the values defined in 4.3 have to be calculated for $0 \leq n \leq N$. The values shown in equation 4.3 are the values used for the message of the fertility node of source word f_j to the hidden variable $y_{(j,i)}$. Therefore, let i_l^* for $1 \leq l \leq I-1$ be all numbers from 1 to I except i. Furthermore let V_c^k be the set of all messages send from the nodes (j, i_1^*) to (j, i_k^*).

$$\begin{aligned}
\alpha_0^j(0) &= 1 \\
\alpha_0^j(n) &= 0 \\
\alpha_k^j(n) &= \sum_{V_c^k:|V_c^k|=n} \prod_{l=1}^{k} n_{(j,i_l^*)\to c}((V_c^k)_{(j,i_l^*)}) \\
&= \alpha_{k-1}^j(n) * n_{(j,i_l^*)\to c}(0) + \alpha_{k-1}^j(n-1) * n_{(j,i*_l)\to c}(1)
\end{aligned} \tag{4.3}$$

Furthermore the probability for a fertility greater than N is needed. This can be calculated in the following way:

$$\begin{aligned}
\alpha_{I-1}^j(N+1) &= \sum_{V_c^k:|V_c^k|>n} \prod_{l=1}^{k} n_{(j,i_l^*)\to c}((V_c^k)_{(j,i_l^*)}) \\
&= 1 - \sum_{n=0}^{N} \alpha_{I-1}^j(n)
\end{aligned} \tag{4.4}$$

Using the precalculated values of α and having only features, which only pay attention if the number of active nodes equals $1, 2, \dots, N$ or is greater than N the messages to the hidden nodes can be calculated in the way described in equation 4.5.

$$\begin{aligned}
m_{c\to(j,i)}(v) &= \sum_{V_c/v} \Phi_c(V_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\
&= \sum_{n=0}^{N} \sum_{V_c/v:|V_c|=n} \Phi_c(V_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\
&+ \sum_{V_c/v:|V_c|>N} \Phi_c(V_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\
&= \sum_{n=0}^{N} \Phi_c(n) \sum_{V_c/v:|V_c|=n} \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\
&+ \Phi_c(N+1) \sum_{V_c/v:|V_c|>N} \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\
&= \sum_{n=0}^{N+1} \Phi_c(n+v)\alpha_{I-1}(n)
\end{aligned} \tag{4.5}$$

4.3.4 First order features

For the first order feature nodes it is again easier to calculate the messages since only 2 hidden nodes are connected to one factored node. So the messages can be calculated straight forward since the sum consists only of 4 parts.

$$\begin{aligned} m_{c\to(j,i)}(v) &= \sum_{V_c/v} \Phi_c(V_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\ &= \Phi(\{v,0\}) * n_{(i',j')\to c}(0) + \Phi(\{v,1\}) * n_{(i',j')\to c}(1) \end{aligned} \tag{4.6}$$

4.3.5 Phrase features

The phrase features can be connected to many other hidden nodes so the straight forward calculation would be inefficient. But the calculation can be done efficiently by using that the potential function of the phrase features can only take two values. The one value is used if the sentence alignment agrees with the alignment of the phrase and the other if this is not the case. Let V_c^* be the allocation of the variables that agrees with the phrase alignment and $v^*_{(j,i)}$ is the value of the variable $x_{(j,i)}$ in this allocation. Furthermore, if this is not the case let $\Phi_c(\overline{V_c^*})$ be the value of the potential function. The two values of the message can then be calculated in the following way:

$$\begin{aligned} m_{c\to(j,i)}(v^*_{(j,i)}) &= \sum_{V_c/v} \Phi_c(V_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\ &= \Phi_c(V_c^*) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v^*_{(j,i)'}) \\ &+ \Phi_c(\overline{V_c^*}) \left(1 - \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v*_{(j,i)'})\right) \\ m_{c\to(j,i)}(1 - v^*_{(j,i)}) &= \sum_{V_c/v} \Phi_c(V_c) \prod_{(j,i)'\in N(c)/(j,i)} n_{(j,i)'\to c}(v') \\ &= \Phi_c(\overline{V_c^*}) \end{aligned} \tag{4.7}$$

To reduce the calculation time for the first value, the following value is precalculated:

$$P_c = \prod_{(j,i)\in N(c)} n_{(j,i)\to c}(v^*_{(j,i)}) \tag{4.8}$$

Then the value can be calculated the following way:

$$\begin{aligned} m_{c\to(j,i)}(v) &= \Phi_c(V_c^*) * P/n_{(j,i)\to c}(v^*_{(j,i)}) \\ &+ \Phi_c(\overline{V_c^*}) \left(1 - P/n_{(j,i)\to c}(v^*_{(j,i)})\right) \end{aligned} \tag{4.9}$$

Then all messages send from one factored node can be calculated in linear time in the number of connected hidden nodes.

4.4 Alignment

After running the belief propagation algorithm for several iterations, the resulting alignment has to be written out. For this, the belief of every hidden node can be used. After running the belief propagation algorithm the belief value can be calculated as

$$b_{(j,i)}(v) = \prod_{c \in N(j,i)} m_{c \to (j,i)}(v) \tag{4.10}$$

Taking the belief as an approximation of the marginal posterior probability, the value of the hidden variable (j, i) can be estimated by taking the most probable value. Consequently, the models will set the link (j, i) only if $b_{(j,i)}(1) > b_{(j,i)}(0)$.

5. Training

The training of the models was done by supervised training on a small hand-aligned development set. For example, in the English-Spanish task it were 100 sentences. In this work different optimization methods were tested. They will be described in the next sections. First, the maximum likelihood (ML) method will be described as this is the most common method for training CRFs. After that, methods that optimize directly towards given evaluation metric will be shown. This will be presented in detail for the alignment error rate in section 5.2. The same principle is then used to train towards an approximation of the F-score introduced in 2.6. Since this is quite similar to the AER-method, only the difference between the methods will be presented.

5.1 Maximum likelihood

In the ML approach the log-likelihood of the development data is maximized. By looking at formula 3.4 this means that the following term needs to be be maximized:

$$\begin{aligned} \sum_{Sentences} logp_\Theta(y|x) &= \sum_{Sentences} log\left(\frac{1}{Z_\Theta(x)}\prod_{c\in C}\Phi_c(V_c)\right) \qquad (5.1)\\ &= \sum_{Sentences}\left(\sum_{c\in C} log\Phi_c(V_c) - logZ_\Theta(x)\right)\\ &= \sum_{Sentences}\left(\sum_{c\in C}\Theta * F_c(V_c) - logZ_\Theta(x)\right)\end{aligned}$$

where a feature value will be set to zero, if it is not defined for a function node c. To find the maximum of the term in 5.1 the gradient descent was applied. Therefore the derivation needed to be calculated. For the first term this is straightforward:

$$\frac{\delta}{\delta\Theta}\sum_{c\in C}\Theta * F_c(V_c) = \sum_{c\in C} F_c(V_c)$$

The derivation of the other term can be calculated in the following way:

$$\begin{aligned}
\frac{\delta}{\delta\Theta} logZ_\Theta(y) &= \frac{1}{Z_\Theta(x)}\frac{\delta}{\delta\Theta}Z_\Theta(x) \\
&= \frac{1}{Z_\Theta(x)}\sum_Y \frac{\delta}{\delta\Theta}\prod_{c\in C} e^{\Theta * F_c(V_c)} \\
&= \frac{1}{Z_\Theta(x)}\sum_Y \sum_{c\in C} e^{\Theta * F_c(V_c)} F_c(V_c) \prod_{c'\in C\setminus\{c\}} e^{\Theta * F'_c(V_{c'})} \\
&= \sum_Y \frac{1}{Z_\Theta(x)}\left(\prod_{c\in C} e^{\Theta * F_c(V_c)}\right)\sum_{c\in C} F_c(V_c) \\
&= \sum_Y p(y|x)\sum_{c\in C} F_c(V_c) \\
&= E_{p(y|x)}\sum_{c\in C} F_c(V_c) \\
&= \sum_{c\in C} E_{p(y|x)} F_c(V_c)
\end{aligned}$$

Consequently, to calculate the derivation, for every factored node c and every example y_k the values of $F_c(V_c)$ and $E_{p(y|x)}F_c(V_c)$ have to be calculated. The first value, $F_c(V_c)$, is the value of the potential function when the random variables have the values that correspond to the reference alignment. So this can be calculated directly. The value of $E_{p(y|x_k)}F_a(\{y\})$ is the expectation value of the potential function and the calculation is more complicated since every possible value of the potential has to be multiplied with the probability that this value is reached. To be able to do this efficiently it has to be done slightly differently for every type of factored node. This will be described in part 5.1.1

After being able to calculate the derivations for all scaling factors the new values of the scaling factors can be updated using a gradient descent as follows:

$$\begin{aligned}
\Theta_{t+1} &= \Theta_t + \eta * \frac{\delta}{\delta\Theta}\sum_{Sentences}(logp_\Theta(y|x)) \\
&= \Theta_t + \eta * \sum_{Sentences}\left(\sum_{c\in C} F_c(V_c) - \sum_{c\in C} E_{p(y|x)}F_c(V_c)\right) \\
&= \Theta_t + \eta * \sum_{Sentences}\left(\sum_{c\in C}\left(F_c(V_c) - E_{p(y|x)}F_c(V_c)\right)\right)
\end{aligned}$$

where η is the learning rate. In most of the experiments the learning rate was set to 0.01/#sentences. If a larger learning rate was selected, the changes of the scaling factors can get to big and the optimization does not convergence. This is especially a problem at the beginning of the training. If a smaller learning rate was selected, the optimization does last to long.

5.1.1 Calculation of the expectation value

For a local feature node the calculation of $E_{p(y|x_k)}F_c(V_c)$ is straight forward since it is only connected to one variable node and consequently $V_c = \{y_{(j,i)}\}$ can only have two different values: link or no link. So it can be calculated as:

$$E_{p(y|x_k)}F_c(V_c) = p(y_{(j,i)} = 0|x) * F_c(0) + p(y_{(j,i))} = 1|x) * F_c(1)$$

where $p(y_a = 0|x)$ is the probability that hidden node $y_{(j,i)}$ equals 0, which can be calculated with the inference algorithm introduced in the previous chapter.
For the fertility feature node the problem is more complicated because there are more variable nodes involved. But the derivation can be calculated efficiently like the messages in the inference because not all possible states have to be considered separately but can be grouped into much fewer states. As mentioned in the last chapter for the feature function $F_c(V_c)$ only the number of active nodes is important. So the probability $p(|V_c| = n|x)$ and $p(|V_c| \geq n|x)$ can be calculated similar to equation 4.3 using dynamic programming. This leads to the following definition:

$$E_{p(y|x_k)}F_c(V_c) = \left(\sum_{0 \leq n < N} p(|V_c| = n|x) * F_c(n)\right) + p(|V_c| \geq N|x) * F_c(N)$$

In this calculation the problem occur that some fertilities get often some probability, but in most of the cases they had not been in the best alignment. In the development of the models another approach to calculate this expectation value was tested and led to better results. The expectation value of a single factored node is set to the potential of the most probable fertility. Since this is done for all fertility nodes, it leads to a good approximation of the expectation value. This results in the following calculation:

$$E_{p(y|x_k)}F_a(V_c) = F_a(argmax_n(p(|\{y_a\}| = n|x), p(|\{y_a\}| \geq N|x)))$$

For the next type of factored nodes, the first order features, the calculation can be done straight forward since there are only two variable nodes involved. The expectation value can then be calculated as follows:

$$\begin{aligned} E_{p(y|x_k)}F_a(V_c) &= p(V_c = (0,0)|x) * F_a((0)) + p(V_c = (1,1)|x) * F_a(2) \\ &+ (p(V_c = (0,1)|x) + p(V_c = (1,0)|x)) * F_a(1) \end{aligned}$$

At last, also the expectation value for factor nodes representing the phrase features have to be calculated. Here, like it was at the local feature nodes, the potential has got only two possible values. Consequently, the expectation value can be determined quickly by calculating the probability like it has been done in the message calculation (equation 4.8):

$$E_{p(y|x_k)}F_c(V_c) = p(V_c = V_c^*|x) * F_a(V_c^*) + p(V_c \neq V_c^*|x) * F_a(\overline{V_c^*})$$

where V_c^* is the variable allocation corresponding to the alignment of the phrase.

5.2 Alignment error rate (AER) optimization

The results of the word alignment are often evaluated using the alignment error rate (AER). Consequently, it would be good, if the parameters could be optimized in a way that the AER is as low as possible. Furthermore, since the ML method maximizes the log-likelihood of a reference alignment, a unique reference alignment is needed. As described in section 2.6 there is not always a unique reference alignment, but some links are marked as "possible". To get a unique alignment for every sentence it has to be decided if all possible links are used as "no links" or as "sure links". So the optimization can make no use of the information given by the "possible links". To be able to optimize towards AER with a gradient descent approach the derivation of the AER with respect to the scaling factors is required. Then the updated scaling factors can be determined as:

$$\Theta_{t+1} = \Theta_t - \eta * \frac{\delta}{\delta\Theta} AER$$

In this case the derivation has to be subtracted because the AER should be minimized and not maximized like the log-probability in the last section.
The problem of building the derivation of the AER is that the AER is not a smooth function so that there does not exist a derivation for all points. But in [SuMI06] and [GWLC06] a method to optimize towards a smoothed F-measure instead of the original F-measure was presented and this led to good results. Since the F-measure and the AER are very similar, this method can also be used to define a smoothed AER and use this to optimize the parameters nearly directly towards the evaluation metric.
The AER is defined as following:

$$\begin{aligned} AER &= 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|} \\ &= 1 - \frac{TP_s + TP_p}{TP_s + FP_s + L_s} \end{aligned}$$

where A is the set of all alignments generated by the word aligner, S and P are the sets of the sure and possible reference alignments. The variable TP_s is the number of true positive sure links that means sure reference links that are also in the generated alignment. Furthermore, FP_s are the false positive sure links, that are links in the alignment, but no sure links in the reference alignment. TP_p and FP_p are defined analogical with the possible links. The variables L_s and L_p are the numbers of sure or possible links respectively. Using this functions and the fact that $TP_s + FP_s = |A| = TP_p + FP_p$, the derivation of the AER can be calculated quite easy:

$$\frac{\delta}{\delta\Theta} AER = \frac{TP_s \frac{\delta}{\delta\Theta} FP_s + TP_p \frac{\delta}{\delta\Theta} FP_p - (FP_s + L_s) \frac{\delta}{\delta\Theta} TP_s - (FP_p + L_p) \frac{\delta}{\delta\Theta} TP_p}{(TP_s + FP_s + L_s)^2}$$

To calculate this derivation the derivations of TP_s,TP_p,FP_s and FP_p need to be determined. Since they are defined as

$$\begin{aligned} TP_X &= \sum_{j,i} 1_{(j,i) \in A} * 1_{(j,i) \in X} \\ FP_X &= \sum_{j,i} 1_{(j,i) \in A} * 1_{(j,i) \notin X} \end{aligned}$$

with $X \in \{S, P\}$ the derivation can be calculated straight forward as:

$$\begin{aligned}\frac{\delta}{\delta\Theta}TP_X &= \sum_{j,i} 1_{(j,i)\in X} * \frac{\delta}{\delta\Theta}1_{(j,i)\in A}\\ \frac{\delta}{\delta\Theta}FP_X &= \sum_{j,i} 1_{(j,i)\notin X} * \frac{\delta}{\delta\Theta}1_{(j,i)\in A}\end{aligned}$$

At this point the original AER cannot be used for the optimization because $\frac{\delta}{\delta\Theta}1_{(j,i)\in A}$ cannot be calculated. Consequently, the parameters can only be optimized towards a smoothed AER, which uses an approximation for this function. Like proposed in [GWLC06] this can be done by using the Sigmoid function σ and the belief value of the hidden node. This leads to the following approximation for the function and the derivation:

$$\begin{aligned}1_{(j,i)\in A} &\approx \sigma_\alpha(b_{j,i} - 0.5)\\ \frac{\delta}{\delta\Theta}\sigma_\alpha(b_{j,i} - 0.5) &= \alpha * \sigma_\alpha(b_{j,i} - 0.5) * (1 - \sigma_\alpha(b_{j,i} - 0.5)) * \frac{\delta}{\delta\Theta}b_{j,i}\end{aligned}$$

Using this approximation, the derivation of the smoothed AER can be determined, if the derivation of $b_{j,i}(v)$ can be calculated. $b_{j,i}(v)$ is defined as:

$$\begin{aligned}b_{j,i}(v) &= \frac{\tilde{b_{j,i}}(v)}{\tilde{b_{j,i}}(0) + \tilde{b_{j,i}}(1)}\\ \tilde{b_{j,i}}(v) &= \prod_{c\in N(j,i)} m_{c\rightarrow(j,i)}(v)\end{aligned}$$

Then the derivation is defined the following way:

$$\begin{aligned}\frac{\delta}{\delta\Theta}b_{j,i}(v) &= \frac{\tilde{b_{j,i}}(1-v)\frac{\delta}{\delta\Theta}\tilde{b_{j,i}}(v) - \tilde{b_{j,i}}(v)\frac{\delta}{\delta\Theta}\tilde{b_{j,i}}(1-v)}{(\tilde{b_{j,i}}(0) + \tilde{b_{j,i}}(1))^2}\\ \frac{\delta}{\delta\Theta}\tilde{b_{j,i}}(v) &= \sum_{c\in N(j,i)} \left(\frac{\delta}{\delta\Theta}m_{c\rightarrow(j,i)}(v)\right) * \prod_{c'\in N(j,i)-\{c\}} m_{c'\rightarrow(j,i)}(v)\end{aligned}$$

Calculating the derivation $\frac{\delta}{\delta\Theta}\tilde{b_{j,i}}(v)$ this way would not be not very efficient. Instead, we use dynamic programming with the following recursion:

$$\begin{aligned}\alpha_0(v) &= 1\\ \alpha_t(v) &= \prod_{k=0}^{t} m_{k\rightarrow(j,i)}(v) = \alpha_{t-1}(v)m_{t\rightarrow(j,i)}(v)\\ \beta_0(v) &= 0\\ \beta_t(v) &= \beta_{t-1}(v) * m_{t\rightarrow(j,i)}(v) + \alpha_{t-1}(v) * \frac{\delta}{\delta\Theta}m_{t\rightarrow(j,i)}(v)\end{aligned}$$

Then the derivation of $\frac{\delta}{\delta\Theta}\tilde{b_{j,i}}(v)$ is equal to $\beta_{|N(j,i)|}(v)$.
To be able to calculate the derivation of $b_{j,i}$ and consequently optimize towards the smoothed AER, the derivation of the outgoing messages of the factored nodes have to be calculated. This will be shown for every type of factored node in the next subsections.

5.2.1 Local feature node

The message a local feature node sends to a hidden node is defined as:

$$\begin{aligned}\overline{m}_{c\rightarrow(j,i)}(y) &= \frac{\Phi_c(y)}{\Phi_c(0)+\Phi_c(1)}\\ \Phi_c(y) &= exp(\Theta * F_c(y))\end{aligned}$$

Then the derivation can be calculated straight forward as:

$$\begin{aligned}\frac{\delta}{\delta\Theta}\Phi_c(y) &= \Phi(y) * F(y)\\ \frac{\delta}{\delta\Theta}m_{c\rightarrow(j,i)}(y) &= m_{c\rightarrow(j,i)}(0) * m_{c\rightarrow(j,i)}(1) * (F_c(y) - F_C(1-y))\end{aligned}$$

5.2.2 Fertility feature node

The calculation for the fertility node is more complex because the outgoing messages also depend on the incoming messages. To keep the complexity low again an approximation is made. In section 4.3.3 the message was defined as:

$$\begin{aligned}\overline{m}_{c\rightarrow(j,i)}(v) &= \frac{m_{c\rightarrow(j,i)}(v)}{m_{c\rightarrow(j,i)}(0)+m_{c\rightarrow(j,i)}(1)}\\ m_{c\rightarrow(j,i)}(v) &= \sum_{n=0}^{N}\Phi(n+v)\alpha_{I-1}(n)\end{aligned}$$

with $\alpha_{I-1}(j)$ defined like in the inference for the fertility features. Using this definition the derivation of the message can be calculated if the derivations $\frac{\delta}{\delta\Theta}\Phi(n+v)$ and $\frac{\delta}{\delta\Theta}\alpha_{I-1}(n)$ are defined.
The first term can be calculated straight forward since $\Phi(j+y)$ is an exponential function. The second term is a sum of a product of incoming messages. The derivation of this can be calculated, if the derivation of the incoming messages is known. But for this calculation the derivation of the incoming messages of the hidden nodes would be needed, which would lead to a long chain of derivations. We therefore approximated the derivation by ignoring the dependency of the scaling factors on all incoming messages. Consequently the derivation of the incoming messages is zero and therefore the derivation of $\alpha(j)$ as well. Using this approximation the derivation for this node can be calculated efficiently.

5.2.3 First order feature node

The first order feature nodes are connected to two other nodes and so the derivation has to be propagated over several iterations to get the exact derivation. To avoid these calculations the same approximation of the derivation is made.
The message is normalized like the one from the fertility node, but the calculation of $m_{c\rightarrow(j,i)}(y)$ is different:

$$m_{c\rightarrow(j,i)}(y) = \Phi(y) * n_{(i',j')\rightarrow c}(0) + \Phi(y+1) * n_{(i',j')\rightarrow c}(1)$$

The calculation of the derivation of Φ can be done since it is an exponential function and the derivation of the incoming messages n are assumed to be zero in the approximation. So the derivation of $m_{c\rightarrow(j,i)}(y)$ can be calculated straight forward.

5.2.4 Phrase feature node

At last, for the phrase feature nodes the derivations of the outgoing messages have to be calculated. The message is again normalized like the last two and the unnormalized message is calculated as described in section 4.3.5 as:

$$\begin{aligned} m_{c\to(j,i)}(v^*_{(j,i)}) &= \Phi_c(V^*_c) * P/n_{(j,i)\to c}(v^*_{(j,i)}) \\ &+ \Phi_c(\overline{V^*_c})\left(1 - P/n_{(j,i)\to c}(v^*_{(j,i)})\right) \\ m_{c\to(j,i)}(1 - v^*_{(j,i)}) &= \Phi_c(\overline{V^*_c}) \end{aligned} \tag{5.2}$$

Using this precalculated value, the derivation can be determined fast although the node can be connected to many nodes. The derivation of Φ can be determined easily since it is an exponential function of the sum of feature values and scaling factors. Since P is a product of incoming messages all other factors of the term have a derivation of zero in the approximation used for the derivation. Consequently the derivation can be determined fast.

5.3 F-score optimization

Although the AER is the most important word alignment metric, there exist other metrics like the F-score. One of the most important advantages of the F-score is that there is a parameter α which balances the trade-off between recall and precision. Optimizing towards this metric enables the word alignment framework to generate more or less links whether the parameter is set more towards recall or precision.
Like for ML and AER optimization the gradient descent should be applied to find scaling factors that lead to a maximal F-score. Consequently, the derivation of the F-score with respect to the scaling factors has to be build. Then the updated scaling factors can be determined as:

$$\Theta_{t+1} = \Theta_t + \eta * \frac{\delta}{\delta\Theta} F\text{-}score$$

To be able to calculate the derivation the F-score using possible links and sure links can be defined the following way:

$$\begin{aligned} F\text{-}score(S, P; A, \alpha) &= \frac{1}{\frac{\alpha}{Precision(P;A)} + \frac{1-\alpha}{Recall(S;A)}} && (5.3) \\ &= \frac{TP_p * TP_s}{\alpha(TP_s + FP_s) * TP_s + (1-\alpha)TP_p * L_s} && (5.4) \end{aligned}$$

with TP_p, TP_s, FP_s and L_s defined as in section 5.2. So the F-score can be written as a function of TP_p, TP_s, FP_s and L_s. To be able to build the derivation of the F-score, the derivation of the four functions have to be defined. Therefore the same approximations as for the AER optimization can be used.

6. Regularization

In this chapter a method to regularize the framework will be presented. First, the problem of overfitting will be discussed in the next section. Afterwards the logarithmic opinion pools of CRFs(LOP-CRFs), which is a parameter-free regularization technique described in 2.5.1, will be used for the CRFs presented in this work.

6.1 Overfitting

Since the CRFs for word alignment have to be trained on hand-aligned data and in most cases only a very small set of hand-aligned data is available, there is always the problem, that there might be overfitting. In the results presented in section 7.8.1 it is shown that this is no big problem, if only a small number of features is used. In contrast, systems using all features could improve a lot on the development data but only a little bit on the test data. So they are the best systems but have the problem of overfitting. If regularization techniques like LOP-CRFs prevent overfitting the performance could be improved even more.

6.2 Logarithmic opinion pools of CRFs (LOP-CRFs)

As already mentioned in section 2.5.1 LOP-CRFs are a good possibility to regularize CRFs because they are parameter-free and easy to incorporate into the CRF framework. The reason for this is that a LOP-CRF is a CRF itself as shown in [SmCO05]. If several CRFs for word alignment are combined into one LOP-CRF, all the CRFs have the same graphical structure. In this case, the LOP-CRF has this structure too. Consequently, if a way to combine the different scaling factors of the CRFs is known, the original framework can be used to calculate the LOP-CRF alignment. So also the belief propagation algorithm can be used.
To see how the scaling factors have to be combined, the LOP probability has to be rewritten as a CRF probability:

$$\begin{aligned}
p_{LOP}(y|x) &= \frac{1}{Z_{LOP}(x)} \prod_{\alpha} \left(p_{\alpha}\left(y|x\right)\right)^{\omega_{\alpha}} \qquad (6.1)\\
&= \frac{1}{Z_{LOP}(x)} \prod_{\alpha} \left(\frac{exp\left(\sum_{c\in C} \Theta * F_c\left(V_c\right)\right)}{Z_{\alpha}\left(x\right)}\right)^{\omega_{\alpha}}\\
&= \frac{1}{Z_{LOP}(x) * \prod_{\alpha} Z_{\alpha}\left(x\right)^{\omega_{\alpha}}} \prod_{c\in C} \prod_{\alpha} \left(exp\left(\Theta * F_c\left(V_c\right)\right)\right)^{\omega_{\alpha}}\\
&= \frac{1}{Z_{LOP}(x) * \prod_{\alpha} Z_{\alpha}\left(x\right)^{\omega_{\alpha}}} \prod_{c\in C} exp((\Theta * F_c\left(V_c\right))_{\alpha} * \omega)
\end{aligned}$$

where $(\Theta * F_c\left(y_c, x_c\right))_{\alpha}$ is a vector of size equal to the number of experts. The entry i for a clique c equals the value of the scalar product of scaling factor vector and feature vector for clique c of expert i. Furthermore, ω is the vector containing all the weights of the experts.
Comparing the last line of the calculation to the definition of the conditional random fields (see equation 2.3) it can be seen that the LOP-CRF is a CRF if $Z_{LOP}(x) * \prod_{\alpha} Z_{\alpha}\left(x\right)^{\omega_{\alpha}}$ equals the normalization factor in equation 2.3.

$$\begin{aligned}
& Z_{LOP}(x) * \prod_{\alpha} Z_{\alpha}\left(x\right)^{\omega_{\alpha}} \qquad (6.2)\\
=& \left(\sum_{y} \prod_{\alpha} \left(\frac{1}{Z_{\alpha}\left(x\right)} \prod_{c\in C} exp\left(\Theta * F_c\left(V_c\right)\right)\right)^{\omega_{\alpha}}\right) \prod_{\alpha} Z_{\alpha}\left(x\right)^{\omega_{\alpha}}\\
=& \sum_{y} \prod_{c\in C} \prod_{\alpha} exp\left(\Theta * F_c\left(V_c\right)\right)^{\omega_{\alpha}}\\
=& \sum_{y} \prod_{c\in C} exp\left(\left(\Theta * F_c\left(V_c\right)\right)_{\alpha} \omega\right)\\
=& Z_{LOP\_CRF}\left(x\right)
\end{aligned}$$

The only difference is that the potential of the factored nodes are calculated differently. Instead of only using the features and scaling factors of one CRF, now the features and scaling factors of all the experts as well as the weights of the different experts have to be used. This is done by using the following formulation:

$$\begin{aligned}
\Phi_c(y_c, x_c) &= exp((\Theta * F_c(V_c))_{\alpha} * \omega) \qquad (6.3)\\
&= exp(\sum_{\alpha} \sum_{k} \theta_k^{\alpha} * f_k^{\alpha}(V_c) * \omega_{\alpha})
\end{aligned}$$

Of course, all methods for making the belief propagation algorithm efficient, like the dynamic programming, can also be applied in this case.

7. Word alignment results

In this chapter the results on different word alignment tasks will be described. First, the data used in the experiments will be presented. Afterwards in section 7.2 the baseline systems will be described. Then the default configuration of the discriminative framework using conditional random fields will be presented and in the following two sections the results using different knowledge sources will be analyzed. Afterwards, some results on the Chinese-English task will be reported.
In section 7.7 the influence of the different features will be discussed. This is followed by a section about the results using the LOP-CRF framework. At the end the results will be compared to the results present by other researches.
If not stated differently, word alignment quality will be measured by the alignment error rate (AER) since this is the most common metric. How alignment quality influences translation quality will be analyzed in the next chapter.

7.1 Data

In this section the data used for the word alignment task will be described. For the word alignment task, two main knowledge sources are needed. First, a small hand-aligned corpus. This is used to train the discriminative framework and to evaluate the word alignment quality. Secondly, a big bilingual corpus is needed. This is used to generate lexica and estimate fertility probabilities. In the experiments, this is done using the GIZA++-Toolkit.
First, the data used for the English-Spanish systems will be described in the next part. In addition, some experiments on the English-French task have been made. The data used for these experiments will be described in section 7.1.2. Afterwards, the data used in the Chinese-English experiments will be described.

7.1.1 English-Spanish

The hand-aligned data used in the English-Spanish experiments were provided by the TALP Research Center. The data is described in [PLbM05]. It consists of a hand-aligned 100 sentences development set and a 400 sentences test set. The sentences are extracted from the Final Edition of the European Parliament Proceedings and contain only sentences with less than 100 words per sentence. The data is annotated with sure and possible links. The test set has 12059 sure links and 5416 possible links.

As additional knowledge source the IBM4-lexica of both directions were trained on the Final Edition of the European Parliament Proceedings provided for the TC-Start 07 Evaluation and the hand-aligned data described before. This corpus consists of 1.4 million sentences. This sentences have got 38 million English words and 40 million Spanish words. The configuration used for the training of the lexica is described in section 7.2 since this was used as a baseline system.

The discriminate word alignment is also able to use features based on Parts-of-Speech(POS)-tags. For the English text, the tags were generated with the Brill Tagger. This tagger generated 40 different tags for the hand-aligned data. The FreeLing tagger was used to generate the tags for the Spanish corpus. The tag set of this tagger is much bigger. Since the tags are only useful if they occur not too rarely on the development set, the tags were grouped by only considering the first two letters of the tag. This can be done, since the POS-Tagset is defined in a way that the first letter describes the tag in the most general way. Then 36 different tags were used for the hand-aligned data.

7.1.2 English-French

The data for the English-French task is from the Canadian Hansard corpus consisting of debates from the Canadian Parliament. Originally, the hand-aligned data consists of a test set with 447 sentence pairs and a development set with 37 sentences. But for the discriminative framework more development data is needed. Therefore, like it was also done by Taskar and others, the first 100 sentences of the test data were used as development data and only the remaining 347 sentences were used as testing data. The original development data is used for the LOP-CRF models, when additional development data is needed. This data was also annotated with possible and sure links, but this time the proportion of both types is different. The original test set of 447 sentences contains 4038 sure links and 13400 possible links.

Again, IBM-lexica were trained on a bigger bilingual corpus. For this task only sentences up to a length of 80 words and with a sentence fertility smaller or equal 8 were used. Then the corpus consists of 1.1 million sentences. In these sentences there are 19 million English words and 22 million French words.

Again, some features based on POS-tags were used. Therefore, the Brill tagger was used for the English side. In this case 41 tags were used for the English hand-aligned data. The French text was tagged using the Stuttgart TreeTagger. This tagger uses 30 different tags for the hand-aligned French data.

7.1.3 Chinese-English

The hand-aligned data for the Chinese-English system is part of the LDC catalogue for the GALE evaluation. It consists of 3401 sentences. After cleaning, 3160 sentences of the data were used. The Chinese words were automatically segmented with the Stanford segmenter. Then the Chinese text contains 70K words and the English side 95K words. In contrast to the other hand-aligned data, this data is only aligned with sure links. The whole set contains 116K links. The data was split into development and test data to be able to use it for the discriminative framework. The first

Table 7.1: Baseline system results for the English-Spanish task

Run	Ps	Rs	Fs	Pp	Rp	Fp	AER
Source-target	75.51	73.82	74.65	83.31	56.21	67.13	21.49
Target-source	79.25	74.63	76.87	87.28	56.72	68.76	19.23
Intersection	**91.93**	66.68	77.29	**96.84**	48.47	64.61	20.64
Union	68.49	**81.77**	74.54	78.22	**64.45**	70.67	20.16
Grow	83.47	74.08	78.50	92.68	56.77	70.41	17.17
Grow-diag	79.35	77.88	**78.61**	89.26	60.46	**72.09**	**16.48**
Grow-final	70.39	80.20	74.98	80.06	62.94	70.48	19.88
Grow-diag-final	71.25	80.51	75.60	80.85	63.05	70.85	19.31
Refined	79.01	78.09	78.55	88.60	60.42	71.85	16.69

2000 sentences were used as test data and the remaining 1401 as development data. To speed up the optimization, often only the first 200 sentences of the development data were used.

The IBM-lexica had to be trained on a bigger bilingual corpus. Therefore, the FBIS corpus was used. The Chinese part was also segmented by the Stanford segmenter. The corpus consists of 341K sentences. Within these sentences there are 9.1M Chinese words and 11.6M English words.

For the hand-aligned corpus as well as for the large corpus POS-tags were generated. The tags of both languages were generated with the Stanford parser. This parser annotates the Chinese text with 34 tags and the English one with 44.

7.2 Baseline system

As baseline system for all language pairs the GIZA++-Toolkit was used. It uses the data described in the last section as training data. Since it is an unsupervised training, the test data can also be used in the training.

The GIZA++-Toolkit was run with a configuration like it is done in the scripts of the Pharaoh-Toolkit. This means that the IBM1 and HMM-models were trained for 5 iterations and the IBM3- and IBM4-models for 3 iterations. This was done for both directions. Then the heuristics described in 2.1 were used to combine both alignments.

The results for the English-Spanish and English-French task are shown in table 7.1 and 7.2. The source-target and target-source results were generated by the IBM4-models. The other results were generated by the heuristics. The results were evaluated in precision(P), recall(R) and F-score(F) on the sure (s) and on the possible (p) links as well as in the alignment error rate (AER).

In both tasks and on both types of links the intersection has the best precision and the union has got the best recall. This is not surprising, since all the heuristics take all the links of the intersection and only select for the links in the difference of the union and the intersection whether to include them into the alignment or not. Consequently, the union has always got the best recall and most of the time the intersection has got the best precision.

In the English-Spanish task the grow-diag heuristic performs best with respect to

Table 7.2: Baseline system results for the English-French task

Run	Ps	Rs	Fs	Pp	Rp	Fp	AER
Source-target	55.89	94.95	70.36	89.18	33.22	48.40	8.6
Target-source	50.59	95.40	66.12	87.35	36.11	51.10	9.86
Intersection	**68.57**	91.97	**78.57**	**96.60**	28.41	43.90	**5.38**
Union	43.85	**98.38**	60.66	83.20	**40.93**	**54.86**	12.12
Grow	54.74	94.69	69.38	92.63	35.13	50.94	6.62
Grow-diag	49.95	96.70	65.87	89.50	37.99	53.34	8.05
Grow-final	45.17	97.44	61.72	84.41	39.92	54.21	11.46
Grow-diag-final	45.41	97.70	62.00	84.73	39.97	54.31	11.16
Refined	50.60	96.08	66.29	89.28	37.17	52.49	8.38

Table 7.3: Baseline system results for the Chinese-English task

System	AER
Source-target	44.94
Target-source	37.43
Grow-diag-final	**35.04**

F-score and AER, followed by the refined heuristic.

In the English-French task the performance is a little bit different from the English-Spanish one. In the F-score with sure links as well as in the AER metric the intersection performs best. Using the F-score on the possible and sure links the union gets the best results. The reason for this seems to be the big number of possible links. The sure links seem to be only the links, which are really clear. Most of these links are already in the intersection, which has a recall of 91.97%. Consequently, adding more links to the alignment the sure F-score only decreases. Regarding all links, the precision of the union is already quite good with 83.20%. So removing links from the union will hurt the F-score calculated on all links.

The baseline results for the Chinese-English task are presented in table 7.3. Here only the grow-diag-final heuristic was used to combine both IBM4-alignments.

The heuristic could improve the word alignment quality. It is surprising that the direction Chinese-English is performing much worse than the other IBM4-alignment. One problem may be the asymmetry of the model. This model cannot align one Chinese word to several English words. If this occurs often this will explain why the word alignment quality is worse for this direction.

7.3 Discriminative word alignment

The different discriminative word alignment systems used in the experiments differ only in some parts. Therefore, first the default system will be described. In the following sections then only the difference to the default system has to be explained. In the next four parts the features used in the systems will be described in detail. First, the set of local features will be presented in 7.3.1. Afterwards, the fertility

features and first order features will be described. Finally, the usage of the phrase features will be explained. In the experiments the features will be divided into two groups: the baseline features and additional features. Therefore, in the following parts, the arrangement of these groups will be explained.
In contrast to the generative approach, the discriminative framework has got many parameters, which have to be trained on a development set. In the development of the system it has be shown that it is good to do this training in three steps. This training process will be described in section 7.3.5.

7.3.1 Local features

The baseline features used in the local feature set are two lexical features based on the IBM4 or IBM1 source-to-target and target-to-source lexica. They were generated by the GIZA++-Toolkit using the setting like the baseline system. Furthermore, for both lexica the normalized source lexical feature and the normalized target lexical feature were used. Additionally, the relative distance feature, the equality feature and the similarity feature were used.
When the link feature was used, it was also part of the baseline features. In this case always two link features for the links generated by the IBM4-alignments of both directions were used. In both cases the weight for every link was set to 1.
In some systems additional features were used. These are the ones based on Parts-of-Speech tags. Therefore, the tags described in section 7.1 were used. Furthermore, the high frequent word features were used as additional features. Here, the 50 most common words in the parallel corpus were used for every language.

7.3.2 Fertility features

The systems used two types of fertility features and both of them belong to the baseline features. First, there are the indicator features as described in section 3.2.2. They were used up to a fertility of 3 and one for all fertilities higher than 3.
In some systems also the GIZA-fertility feature was used. This used the fertility probabilities generated by the GIZA++-Toolkit. They were generated with the same configuration as the lexica. The GIZA-feature was also used up to a fertility of 3.

7.3.3 First order features

In the group of first order features only features, which indicate that both links are active were used. The other two features did not improve the systems. If not stated differently, the following four directions were used: (1,1), (2,1), (1,2) and (1,-1). Consequently, the first order features add four more features to the baseline group. POS-based first order features are also used as additional features in some systems. The features based on the starting word pair of a link were used.

7.3.4 Phrase features

In the default system no phrase features were used because the corpus has to be aligned before the phrases or groups can be extracted. If the features were used, they belonged to the group of baseline features. Both types of phrase features were used with the feature values described in section 3.2.4. They were used in the exact mode where only the links of the phrases are allowed to be active and in the normal mode, where the phrase only indicated that all links of the phrase are active. The phrase features were used with a maximal phrase length of four.

Table 7.4: Results using different optimization methods

Step	Without link feature	With link feature
Step 1 ML	20.32	18.57
Step 1 AER	21.28	16.88
Step 2 ML->AER	**18.12**	**16.09**
Step 2 AER->ML	19.84	18.16

7.3.5 Training

The training was done in three steps since this led to the best systems. In every step the best scaling factors according to the optimization metric were used as initial values for the next iteration.

In the first step only the baseline features were used and the system was optimized with the maximum likelihood method. As reference only the sure links were used since the ML-optimization needs a unique reference and using the sure links only produced better results than using both links as reference. The optimization uses a learning rate of 0.01 for every feature, because for bigger values the changes in every iteration had been too big. This was done for 200 iterations.

In the next step again only the baseline features were used, but the system was optimized towards the AER. In this case the original reference with sure and possible links could be used. This optimization was done with a learning rate of 1 for 200 iterations.

This two-step approach has been selected because its performance was the best in the experiments. It is illustrated in the two examples in table 7.4. The two system have the same features as the default system described above using the IBM4-lexica and the GIZA-fertilities. One systems uses the link feature, the other one does not. It can be seen, that one time the AER-optimization is better in step 1 and one time the maximum likelihood version is the better one. But the best results were obtained by combining both techniques in the order ML-optimization and then AER-optimization.

A reason for this could be, that the ML-optimization is searching more globally and the AER-optimization is searching more exactly, but more locally. The AER-optimization could be more exact, because it can make use of sure and possible links. This would explain, why it is better first to do the ML-optimization and afterwards the AER-optimization. Furthermore, it could explain, why the AER is better when using the link feature. If the link feature is used, this is a strong evidence for a good alignment and this leads initially to good alignments. Consequently, to get really good ones the other parameters have only to vary a little bit. In contrast, the ML approach would search more globally and cannot improve that much of this feature.

In the last step the scaling factors for the baseline features were not changed and only the additional features were optimized. In this step the AER-optimization method has been applied. The additional features were optimized separately since they have a stronger tendency towards overfitting and they occur more rarely than the other features. Consequently, it would be complex to find a learning rate for the different features so that they are all optimized in the same speed. To avoid this, they were not optimized at the same time. A problem could be, that in this case, no global optimum is found. But if the baseline features are optimized again in a fourth step, this does not improve the system.

The behavior of one system was different. The English-Spanish system using the IBM4-lexica and GIZA-fertilities described in section 7.4 improved, if in the last step all parameters were optimized and not only the one of the additional features. But since this method did only help in this configuration, it was only used in this case.

7.4 English-Spanish

In this section the results for the English-Spanish systems using different knowledge sources will be described. As knowledge source different outputs of the GIZA++-Toolkit were used as well as phrase tables generated with the discriminative word alignment. The results are summarized in table 7.5. The results during the different steps of the optimization are shown. In step 3(a) only the high frequent word features were used as additional features. In step 3(b) also the POS-based features were used. This was done since the POS-tags represent an additional knowledge source.

The first system using the fewest information is the system called "IBM1". The only knowledge sources for this system are the IBM1-lexica of both directions. Consequently, it uses no GIZA-fertility feature and no link feature. This system is better than the GIZA-alignment from source to target, but in contrast to this all other baseline results are better than this. Consequently, using only the IBM1-lexica does not seem to give a good alignment, but of course it has the advantage, that the IBM1-lexica can be generated quite fast.

The next system uses IBM4-lexica instead of a IBM1-lexica. This is the only difference between the systems. This improves the system by 2.2 AER points after the last step. Of course, then the high-frequent word features do not improve the system as much as in the first system. If adding also the POS-tags, the system is in the same range as the heuristics used to combine the GIZA-alignments.

The fertility probabilities were the next knowledge source that was added. The system using this information is called "IBM4+GIZA-fert.". If it is not used together with the POS-tags the system performance improved by around 0.6 AER points. As already mentioned in the section 7.3.5, this system will improve, if in step 3 all features are trained. Then also a AER of 17.82 is reached. So if the GIZA++-fertilities are used together with the POS-tags the performance cannot be improved, but at least it is not worse than the one using no GIZA++-fertilities.

In the "link feature" system in addition to the knowledge sources used in the last system, the links of both IBM4-alignments are used as features. This improves the system by 2 AER points and the system using POS-tags even by 3. Using these features the discriminative word alignment is better than all heuristic ones by at least 1 AER point. This improvement is no big surprise since the IBM4-alignments are already quite good ones and so the links give strong evidence for a good alignment. But it is important that the quality of the generated alignment is by far better than the one of the input. The IBM4-alignments have got an alignment error rate of 21.49 and 19.23, but the resulting discriminative word alignment generates an alignment with an AER of 15.36. So the alignment error rate is 4 AER points or 20% better than the best alignment that is used as input.

In the last system the whole training corpus was aligned with the "link feature" system. Then the phrase features using the groups of this alignment were added to

Figure 7.1: Comparison of alignments

	para	la	televisión	pública
to	BD			
public		B		D
service			B	B
broadcasting		▨	BD	

(a) Example 1

	,	pero	son	importantes
but		BD		
they	B		D	
are			BD	
still			B	
important				BD

(b) Example 2

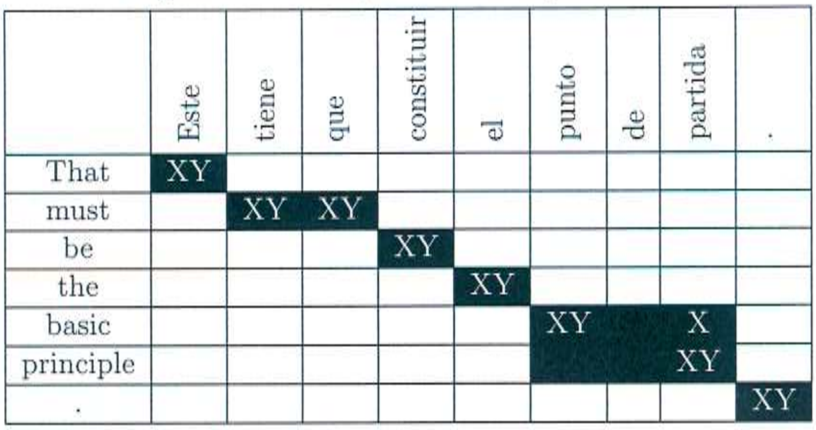
the system. These additional features could improve the system by 0.6 AER points and led to the best results of the discriminative word alignment with an alignment error rate of 14.77.

In figure 7.1 two alignments of the best baseline system and the best discriminative word alignment system are shown. A black cell represents a sure link and a gray cell a possible link. The links set by the baseline system are marked by a B and the ones of the discriminative framework by a D. The first example is part of the sentence pair "*Secondly, the same European rules should apply to public service broadcasting and to commercial stations offering capacity for public services.*" and "*En segundo lugar, las mismas reglas europeas para la televisión pública y las cadenas comerciales que ofrecen suficiente espacio para servicios públicos.*". In this example, the baseline system has more problems when aligning words in non-monotone order. This is especially a problem for the baseline system in case of very short words. There the discriminative word alignment is often better than the baseline system. This is also shown in the second example which is part of the sentence pair "*These measures may seem small but they are still important to our common goal of mobility within the internal market.*" and "*Estas medidas pueden parecer ínfimas, pero son importantes bajo la perspectiva de nuestro objetivo común de la libre circulación dentro del mercado interno.*".

The results of these experiments showed also, that the POS-tags seemed to be a valuable knowledge source. For all different knowledge sources the systems in step 3(a) and step 3(b) differ only in the use of the POS-features. If all these systems are compared, the POS-features can mostly improve the system performance. In all systems except the "IBM4+GIZA-fert" this leads to a system that is at least 0.6 AER points better than the one using no POS-tags.

One example of an improvement by the POS-tags is presented in figure 7.2 that shows an alignment of a sentence pair by the "link feature" system with and without the POS-tags. The alignment is shown like in figure 7.1. The X's show the links of the system with POS-tags and the Y's the ones of the system using no POS-tags. They differ only in one link, but especially for the translation task this is an important one. In the POS-alignment no sub-phrase of the phrase pair "basic principle"

Figure 7.2: Example of POS improvement



	Este	tiene	que	constituir	el	punto	de	partida	.
That	XY								
must		XY	XY						
be				XY					
the					XY				
basic						XY	■	X	
principle						■	■	XY	
.									XY

Table 7.5: Results for discriminative word alignment on EN-ES task using different knowledge sources

Name	Step 1	Step 2	Step 3(a)	Step 3(b)
IBM1	24.21	21.81	20.82	
IBM4	21.72	19.05	18.67	17.82
IBM4+GIZA-fert.	20.32	18.12	18.02	18.53
Link feature	18.57	16.09	15.97	15.36
Phrase feature	18.31	15.50		**14.77**

"punto de partida“ can be extracted like described by the reference alignment. But the alignment using no POS-tags allow several sub-phrase pairs. The direct influence of the POS-tags can be seen by the scaling factor of the POS-tag pair of the words in the additional link. This scaling factor has got a positive value of 0.41.

In conclusion, every knowledge source that is available should be used. It nearly always improved the system to add further information to the word alignment. This clarifies, how important the possibility of the discriminative word alignment is to add easily a high number of features, since this enables the system to use all available knowledge sources. Furthermore, the discriminative word alignment system could reach an AER of 14.77. In contrast, the best heuristic generates an alignment with an AER of 16.48. So the discriminative word alignment could improve the word alignment quality by 1.71 AER points or 10%.

7.5 English-French

The results for the English-French task using the discriminative word alignment with different knowledge sources are shown in table 7.6. Again the results during the 3 steps of the training are shown. In step 3 the systems using the IBM1-lexica do not include POS-based features, but the others do.

As the only knowledge source the first system "IBM1" uses the IBM1-lexica of both directions. In this task this system is already better than all heuristics. Only the intersection of both IBM4-alignments is better. One reason may be, that the hand-alignment has many possible links and the discriminative word alignment can better

Table 7.6: Results for discriminative word alignment on EN-FR task

Name	Step 1	Step 2	Step 3
IBM1	11.14	6.37	5.83
IBM1+phrase feature	10.69	6.28	5.69
IBM4	10.51	6.16	5.93
IBM4+GIZA-fert	9.72	6.24	5.77
Link feature	7.65	4.54	4.60
Phrase feature	7.54	4.39	**4.30**

be adapted to this alignment.

For the "IBM1+ phrase feature" system the whole corpus has been aligned with the "IBM1" system and the links made by this system were used to create a group table. Using this additional knowledge source, the AER could be improved by 0.2 points.

The "IBM4" system is nearly the same as the "IBM1" system except that it uses the IBM4-lexica instead of the IBM1-lexica. In contrast to the English-Spanish system this system does not perform much better than the one that uses only the IBM1-lexica. After the last optimization step its performance is even worse, but these configurations are not directly comparable since the "IBM4" system also uses the POS-tags. Maybe there are again some problems using this information.

In the system "IBM4+GIZA-fert" the GIZA++-fertilities are also used as an knowledge source in the GIZA-fertility feature. In most configurations the additional knowledge source improves the system performance. But again the improvement is not very large.

After that the IBM4-alignments generated by the GIZA++-Toolkit were added to the system. The resulting system is call "link feature". As already seen in the other task, this feature does improve the system performance a lot and the system does perform better than all baseline systems. This system generates an alignment with an AER of 4.60, which is also much better than the two alignments used as input and having an AER of 8.6 and 9.86.

Finally, the "phrases feature" system uses, in addition to this, the phrase features generated by the alignment of the last system. Therefore, the whole training corpus was aligned with the "link feature" system and the groups in this alignment were extracted and used as features. These features could improve the system by around 0.3 AER points leading to the best result of an alignment error rate of 4.30.

For the English-French task additional information used in the discriminative framework could improve the alignment quality as well. By using all this knowledge sources , this leads to better alignments than the one generated by the generative approach. In the end an alignment error rate of 4.3 was reached, which is 1.08 AER points or 20% better than the best baseline system.

7.6 Chinese-English

In addition to the two European language pairs, the discriminative word alignment was evaluated on the Chinese-English task. The results of this task are shown in table 7.7. In the first two steps as well as in version a of step 3 the systems were optimized only on the first 200 sentences of the development data. The third step was also trained on the whole development set. The results of these configurations are shown in version b of step 3. The main disadvantage of the second version is that the training lasts longer.

In the first system "default" all available information have been used and the system is configured like the systems used for the last language pairs. This system uses both IBM4-lexica, the fertilities generated by the GIZA++-Toolkit and the links of both IBM4-alignments. This first system is already 4 AER points better than the combined IBM4-alignments.

In the next system ,"no similarity", the similarity feature was not used. This seems to improve the system since this information is not useful for language pairs that are very different. The alignment error rate could be improved by 0.8 points by removing this information. If the additional development data is used in the third step an AER of 29.66 was reached.

This language pair has got many more $1 : n$ and $n : 1$ links than word alignments between European languages. Especially consecutive words are often aligned to the same word in the other language. So in the "add. directions" system the directions (1,0) and (0,1) were added to the last system to be able to model this behavior in a better way. In contrast to the experiments on the English-Spanish task in section 7.7.3, where additional directions could not improve the quality very much, in this case the AER could be decreased by more than 2 points. The additional directions will be especially helpful if the additional features were used, too.

Then the whole training corpus was aligned with this alignment as well as with a system that has the same configuration but is optimized towards a F-score with an α-value of 0.7. So the second system is focusing more on the precision than the first one. After that group tables are created from these alignments and a word alignment which uses this table as additional knowledge source is generated. Both configurations could improve the system, but the best results were obtained with the high precision group table and the best AER of 26.90 was reached. This is an improvement of 8.14 points or 23%.

The additional features even seem to be more important for this task than for the last language pairs. In the "add. direction" system the performance could be improved by more than 4 AER by using these additional features. So this information seems to be one important point why the discriminative word alignment models are better than the baseline system. So it demonstrates again, how important it is to use all available information to generate a good alignment.

Another point that it different to the other language pairs is, that more hand-aligned data is available. This leads to a longer training, but the word alignment could be improved. The training on the large development set in the last iteration could improve the performance by about 0.6 AER points.

7.7 Features

In this section the influence of the different features will be described. First, experiments concerning the local features will be shown. Afterwards, in section 7.7.2, the different types of fertility features will be examined. In the following parts the first order features and the phrase features will be tested.

All experiments were done on the English-Spanish task. The default configuration

Table 7.7: Results for discriminative word alignment on the CH-EN task

Name	Step 1	Step 2	Step 3(a)	Step 3(b)
Default	36.40	33.71	30.97	
No similarity	34.81	33.48	30.24	29.66
Add. directions	33.74	32.16	27.96	27.26
Phrase feature	32.65	30.23	27.71	27.00
Phrase feature(high P.)	32.28	30.51		**26.90**

for all the experiments uses the IBM4-model lexica and the GIZA-fertility feature if not stated differently. Most of the experiments were done with and without the link feature since this feature is a very strong indicator. Consequently, if this feature is used the influence of all other features is not that big. If no link feature was used, in the third step of the optimization all parameters were optimized as explained in section 7.4.

7.7.1 Local features

The lexical probabilities are a very important local feature. But since some information about the probability of a word translated into another is needed, no experiments were done without using lexical information. But the influence of different lexica has already been examined in the section about the different knowledge sources. There it has been shown, that for the Spanish-English task, the system performance could be improved by 2 AER points if an IBM4-lexica is used instead of an IBM1-lexica. Consequently, the lexical information seemed to be a quite important information and the quality of these lexica influences directly the quality of the alignment.

The influence of the other 4 types of local features, which are part of the baseline features, is shown in figure 7.3. It has been evaluated on four different configurations. Two configurations use only the baseline features (Base) and two use also the additional ones (add.). Furthermore the features were evaluated with (+ LF) and without a simultaneous use of the link feature. The influence was evaluated by comparing the performance of the default systems to the systems that do not use this feature. In the figure the difference between the alignment error rate of the system without the feature and the default system is displayed. Consequently, if the value is high, this feature has got a large positive influence on the alignment quality.

The most important features in these experiments were the normalized lexica. These are four features for source and target normalized probabilities of both lexica. Adding these features improved the performance of the default system by 8.62 alignment error points, which are 32%. Especially, if the other features do not give a good clue for a good alignment these features improve the alignment quality. One main advantage may be, that they compare the translation probability to all other source links or target links. So they are able to find the alignment for words were all lexical probabilities are quite bad, for example, if a word has got many possible translations, but in this sentence only one occur.

The next feature is the relative distance feature. Using this feature does always improve the alignment quality, too, but not as much as the normalized lexical features

Figure 7.3: Influence of different local features

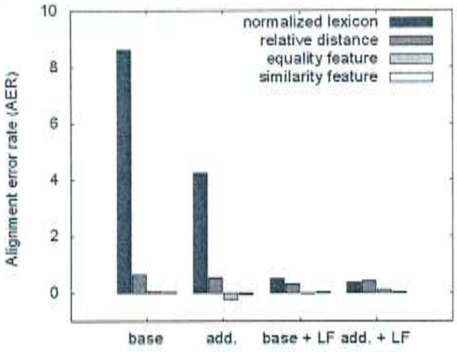
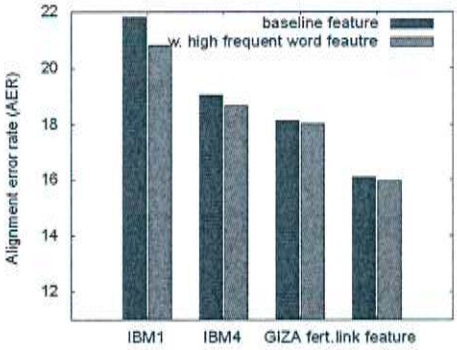


Figure 7.4: Influence of the high frequent word features



do. But it is very interesting that this feature seems to improve all systems nearly equally and the influence does not depend that much on the quality of the other features. So it is even more important in the last configuration using all features than the normalized ones.

Furthermore the influence of the similarity and equality features is displayed in the figure. The influence of these features is not that big and in some configurations they even seem to hurt the performance. The reason for this is that the features are only interesting for quite a small number of links. For example, the equality feature is only important for links between numbers or names, but most of the words are neither numbers nor names. In contrast, the relative distance is important for every possible link.

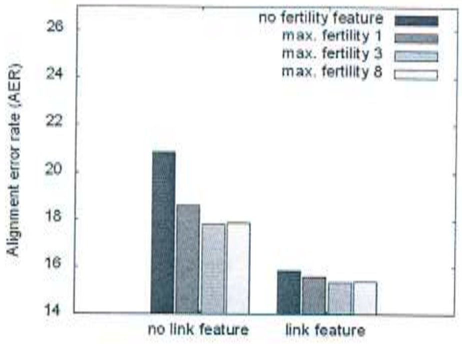
The influence of the POS-based features has already been discussed in the last section as they just described two additional knowledge sources. But the influence of the high frequent word features is displayed in picture 7.4. There different systems using only the baseline features were compared to the same system using also the high frequent word features.

The influence of these features are not as big as other features, but all systems could be improved by using these features. Again, the system, in which the other features have less information, could benefit more from these features.

7.7.2 Fertility features

In this part the different fertility features will be evaluated. First, experiments concerning the maximal considered fertility will be done. Afterwards the GIZA-fertility

Figure 7.5: Results using different maximal fertilities



feature will be evaluated.

In the discriminative word alignment models all fertilities greater than a given N are not distinguished. This has to be done to limit the calculation time. In the default system $N = 3$ is selected. In the experiments shown in figure 7.5 the impact on the word alignment of this decision has been evaluated.

Therefore, four systems are compared. The first one does not use fertility features. The others use a maximal fertility of 1, 3 and 8. As shown in the figure, using no fertility features does hurt the performance. They are not as important as the local features or the first order features, but they can improve the system by more than 2 AER points in the case of no link feature and 0.5 if the link feature is used.

If the maximal fertility grows from 1 to 3 this will improve the word alignment quality, but the difference is not as big as the difference between no fertility features and a maximal fertility of 1. So the difference is between 0.8 and 0.1 AER points. Increasing the maximal fertilities even more towards 8 does not improve the performance. The main reason will be that there are nearly no words which have a fertility bigger than 3 and consequently it does not matter how many more fertilities are considered.

To be able to calculate the alignment in an adequate amount of time, some approximations had to be made. As described in section 4.3.3 only features, that consider the probability up to a given maximal fertility, can be used. These experiments show that at least for this task, this constrained to the type of possible features does not hurt the word alignment performance. Even if it would be able for the models to use features considering all possible fertilities, as well, most probably the alignment quality would not be better.

In addition, the impact of the GIZA-fertility feature was evaluated. Therefore, four different models were evaluated once with the GIZA-fertility feature and once without this feature. The results are shown in picture 7.6.

For all systems except the one using the additional features but not the link feature, the GIZA-feature could improve the performance. In these three systems the alignment error rate could be decreased by 0.2 to 0.8 AER points. Consequently, as already mentioned in the section about the knowledge sources, the GIZA-fertilities seemed to be a feature that could help to improve the quality of the generated alignment.

Figure 7.6: Results concerning the GIZA-fertility feature

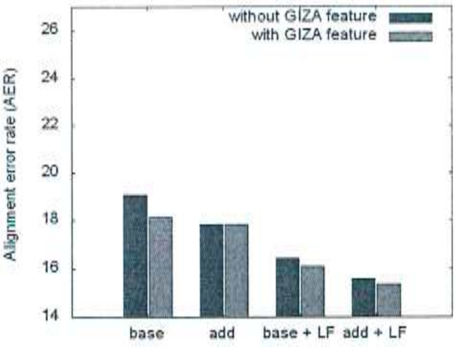
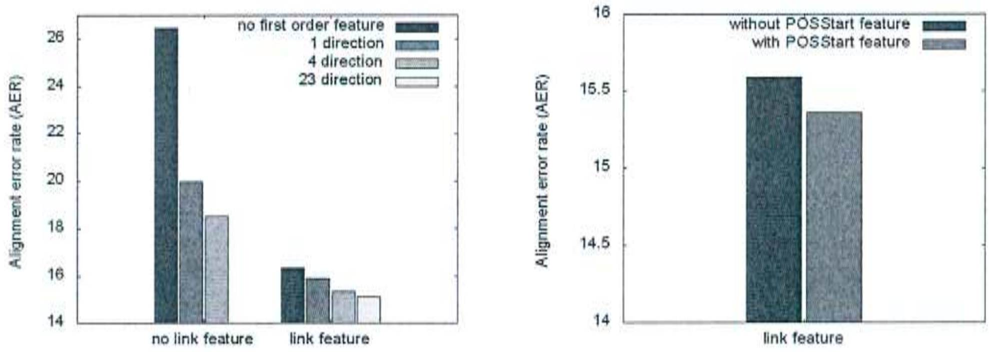


Figure 7.7: Results for First order features



(a) Different directions

(b) POSStart features

7.7.3 First order features

The first experiments concerning the first order features should explore the influence of the directions used in the first order features. Therefore, four different configurations were evaluated. The first one uses no first order features, the other use 1, 4 and 23 different directions in the first order features.

The system using only one direction uses the (1,1) direction, since this should be the most important one. The directions (1,1), (2,1), (1,2) and (1,-1) are used in the system with four directions as it is done in the default system. The system with 23 directions uses different long directions forward and backward. It has also the directions (0,y) and (x,0) to model the dependence between links to the same target or to the same source word. The results for the systems with and without link feature are shown in picture 7.7(a).

It is shown in the picture that the first order features are quite important ones. In the case with no link feature the performance could be improved by 6.5 AER points by just adding a first order feature with one direction to a system with no first order feature.

Furthermore, the experiments showed that by adding further directions to the first order features the performance has been improved. In both cases the AER decreases by adding further directions. Of course, the improvement decreases by adding more and more features. That is the reason why in most experiments four directions were used, since the calculation time grows linear with the number of directions.

Figure 7.8: Example of first order features

	la	primera	semana	de	vida	del	Tratado	de	Amsterdam	.
the	YZ									
first		XYZ								
week			XYZ							
of				YZ						
the				Y	▨					
life					XYZ					
of						YZ				
the						Z		Y		
Amsterdam								▨	XYZ	
Treaty							XYZ			
.										XYZ

Furthermore, the difference between the different numbers of directions is smaller when using the link feature, since this is already a strong evidence for where a link should be.

In figure 7.8 example alignments of the systems using no link feature are shown. The shown phrase pair is part of the sentence pair "*It is a coincidence, but perhaps a fortunate one, that this very week when we are in the last week of the life of this House, we are also in the first week of the life of the Amsterdam Treaty.*" and "*Es una coincidencia, aunque quizá una feliz coincidencia, que esta misma semana, que es la última de la legislatura de esta Asamblea, sea también la primera semana de vida del Tratado de Amsterdam.*".Again, the sure links are shown as black cells and the possible links as gray ones. The links set by the system using no first order feature are marked by a X. The Y's represented the links by the model using only the direction (1,1) and the Z's the ones of the system using 4 directions.

The system with no first order feature has big problems aligning the articles since here the lexical features give no good clues like for other words and it does not use the first oder dependences to the other words. In addition, this system aligns words where no links are near this link. For example, the Spanish word "vida" in the phrase is aligned into the word "life", which is in the English sentence but not in the presented phrase. The system using one direction can model this better, but has problems if the first oder dependency has not the direction (1,1). Furthermore, it sometimes overestimates the first oder dependency in this directions as in the case of the link between the last "de" and "the".

In a second group of experiments the influence of the POS-based first order features were evaluated. These experiments were evaluated on the default system using only four directions and the link feature. As it is shown in figure 7.7(b) these features improve the system performance, but they are not that important.

Table 7.8: Results for discriminative word alignment on EN-ES task using the phrase features

Name	Baseline	Add.feature
No phrase feature	16.09	15.36
Phrases	16.00	15.38
Exact phrases	15.83	15.38
Groups	15.50	**14.77**
Exact groups	15.67	14.99

7.7.4 Phrase features

To use the phrase features, first the whole corpus of 1.1 M sentences has to be aligned. Afterwards, using this alignment phrases or groups, as described in section 3.2.4, have to be extracted and a phrase table is build. The alignment of the corpus was done with the "link feature" configuration described in the last section. Then four different types of phrase features were used in the experiments. The phrase features were added to the "link feature" system. The results for these experiments are shown in table 7.8.

The first system "phrases" uses phrases extracted from the corpus as described in section 3.2.4. The performance of the system using only the baseline features could improved slightly, but the alignment of the system using all features could not be improved.

The second system , "exact phrases", uses the same phrase table as the first one, but this time the phrase features do not only look at the links set active but at all links involving a word of the phrase. In the case of the phrase "the Eurozone ", "la zona euro" , 1-1 2-2 2-3, for example, for the first system the phrase features would be active if in addition to the links in the phrase "la" is also aligned to "Eurozone". This is not the case in the second configuration. The disadvantage of this configuration is, that there are many more edges in the graph and consequently the alignment takes longer. As it can be seen in the table 7.8 these phrase features improve the system using only the baseline features, but not the one using also the additional features.

The next system "groups" uses groups like they are described in section 3.2.4. These features improve the system using only the baseline features and the one using additional features and leads to the best result of an AER of 14.77.

The last system , the "exact groups", uses the same groups as the last system, but it does this in the "exact mode" like described for the "exact phrases" system. This time the results using the "exact mode" are worse than the ones using the "normal mode".

In conclusion, the best results were obtained using the group table. The phrase table could not improve the alignment quality of this task. Furthermore the exact mode does not improve the alignment quality although the calculation time does increase dramatically. The system performance of the "link feature" system could be improved by the phrase features from 15.36 to 14.77. This is an improvement of 3.8%. The disadvantages of these features are that the whole corpus has to be aligned and that the calculation time increases.

7.7.5 Conclusion

In the analysis in this section, it was shown, that nearly all features help to improve the system. Only for a few features it is not clear, if they could really help to improve the word alignment quality. Furthermore, the most important features seem to be the local features. Here, the normalized lexical features seemed to help a lot to find a good alignment. Furthermore, the relative distance feature can give advice for a good alignment.

The next important group of features are the first order features. For the selected task they were more important than the fertility features. Adding more directions could reduce the AER, but the reduction is getting less the more directions are already added. Since the calculation time is growing with the number of directions a trade-off has to be made.

In contrast to the first order features the alignment quality does not improve by considering more and more fertilities. Up to the fertility four an improvement could be seen, but then no improvement towards a maximal fertility of nine is recognizable. Consequently, the constraint only to consider the fertilities up to a given fertility seems not to hurt the system performance.

Both types of additional features, the POS-based and the high frequent word features, could reduce the alignment error rate, but in most of the systems the POS-based ones are more important than the high frequented ones.

At last, the phrase features could encode additional information and so improve the alignment quality. The best results were obtained when using groups in "normal mode". The disadvantage of these features is, that they lead to additional calculation time and the whole training corpus has to be aligned first.

7.8 LOP-CRF word alignment

The LOP-CRFs were introduced in section 6 as a possibility to prevent overfitting in conditional random fields. In this framework, different CRFs were combined to build a more general CRF. In this section this approach is applied to discriminative word alignment.

To see if a regularization method is needed, first the overfitting using the CRF word alignment will be examined. Afterwards, the general configuration of the experiments will be described. In sections 7.8.3 and 7.8.4 the results for the different language pairs will be discusses.

7.8.1 Overfitting

In this section the overfitting of the discriminative word alignment using conditional random fields will be discussed. The performance on the development and test set is evaluated to see if overfitting exists. The results for the different systems are displayed in table 7.9.

The first four systems are different heuristics to combine both GIZA-alignments. Since they are trained unsupervised, these systems have no overfitting effect. So they can be used to compare the difficulty of both sets. As the results show, in the English-Spanish task the development set seems to be an easier task than the test set. The AER difference is between 20 and 26%. In the French-English task, both set seemed to be equally difficult. Some heuristics are better on the development set

Table 7.9: Results for development and test set

	EN-ES			EN-FR		
Name	Dev. set	Test set	Difference	Dev. set	Test set	Difference
Grow-diag-final	14.74	19.31	24%	10.91	11.16	2%
Refined	12.96	16.69	22%	8.32	8.38	1%
Union	15.31	20.16	26%	11.52	12.12	5%
Intersection	16.59	20.64	20%	6.05	5.38	-12%
Baseline features	12.80	16.09	20%	4.8	4.54	-6%
All features	9.21	15.36	40%	3.5	4.6	24%

and some on the test set.
The next systems are the "link feature" systems described in section 7.4 and 7.5 once only using the baseline features and once also using the additional features. For the systems using only the baseline features overfitting seems to be no problem, since the difference in the performance on both sets is similar to the one using the heuristics. But if the additional features were added, the performance on the development set increases dramatically, but the one on the test set increases only a little bit. Consequently, the difference increases and overfitting seems to become a problem.

7.8.2 Experiments

Experiments using the LOP-CRF framework were done for the English-Spanish task and for the English-French task. The problem of the LOP-CRFs is, that additional development data is needed to optimized the weights of the different experts. This data has to be different from the one on which the CRF systems used in the LOP-CRFs are trained. Since no additional hand-aligned data for the Spanish-English task is available, the development set had to be divided into two parts of 50 sentences each. Consequently, the CRFs used in the framework could only be trained on a development set of 50 sentences. That's the reason why the results for these systems are worse than the results presented in the last sections. For the English-French task the 37 sentences of additional development data that were not used in the experiments are used for the LOP-CRFs.
In the experiments it was shown, that the word alignment is not very sensitive to small changes of the expert weights. Because of that only multiples of 1/6 were used as weights for the experts. Since only up to four experts were used in the experiments and the sum of all weights had to be 1, only up to 84 combinations are possible. So in the training all possible combinations could be tested and the best combination on the development set was selected. These weights were than used to align the test set and evaluate the performance.

7.8.3 English-Spanish system

The results for the English-Spanish LOP-CRF system are shown in table 7.10. For every system, first the results of the experts are presented and then the result of the

Table 7.10: Results for LOP-CRF word alignment (EN-ES)

Name	Exp 1	Exp 2	Exp 3	Exp 4	Test
Base + add.	16.23	16.01			15.86
Phrase features	16.01	15.79	16.02	15.22	**15.14**
Different metrics (base.)	16.23	16.23	17.00		16.30
Different metrics	16.01	16.56	16.81		15.71

LOP-CRF ,build by these experts, is shown. As mention in the beginning, the results of the experts are sometimes worse than the results presented by these systems before because they are only trained on 50 sentences and not on the whole development set. The default system in this experiments was the "link feature" system.

The first system "base + add." combines a system using only the baseline features and one also using the additional features. Both were trained on the first part of the development data and the LOP-CRF on the second part. As shown in the table, the LOP-CRF performed better than the single experts, but it could not outperform a CRF word alignment, which was trained on the whole test set. This reached a AER of 15.36.

In the next system ,"phrase features", the default system is combined with three systems using the phrase features in "normal mode" and "exact mode" and the group features. The resulting system is better than the best single system, but only a little bit.

Both "different metrics" systems combine the default system, which is optimized towards AER, with two system optimized towards the F-score. One system is optimized towards a F-score of 0.3 and is consequently more concentrating on the recall and the other is trained with an α-value of 0.7 and is more looking towards precision. Again, all systems are trained on the first part of the development data and the resulting LOP-CRF system on the second part. The LOP-CRF system using only the baseline features does not improve. This also indicates that there is no overfitting using only the baseline features and consequently regularization techniques do not help to improve the system. In contrast, the system using all features does improve by 0.3 AER points. But again, a single system trained on all development data performed better.

7.8.4 English-French system

For the English-French task some experiments with the LOP-CRF word alignment have been made, too. The results for these experiments are displayed in table 7.11. The default system in these experiments was the "link feature" system used already in section 7.5.

In the first system "base + add." combines the default system with and without additional features. The LOP-CRF system was optimized on the original development set of 37 sentences. This system outperformed both single system, but is only slightly better.

In a second system "phrase features" the default system is combined with a system using the phrase features with a group table. This phrase feature system reached the best results of all CRF system. In this case the combination of both systems

Table 7.11: Results for LOP-CRF word alignment (EN-FR)

Name	Exp 1	Exp 2	Exp 3	Test
Base + add.	4.54	4.60		4.46
Phrase features	4.60	**4.30**		**4.30**
Different metrics	4.60	6.20	4.62	4.52

could not improve the performance.
In addition the default system was combined with two systems optimized towards the F-score with α-values of 0.3 and 0.7. Here the LOP-CRF combination could again improve the word alignment quality. But, as it occurred in the first system, the improvement is very small.

7.8.5 Conclusion

The LOP-CRF framework can be used to improve the word alignment quality if the single systems have problems with overfitting. This is especially the case if the additional features are used. Then the AER can be decreased slightly by using this framework. This can be done by combining system with different features or with different optimization techniques.
The main problem is, that additional training data is needed. For the English-Spanish system it was better to train a single system on the whole development data than splitting this data and train the single CRFs only on one part of the data. Furthermore, the training time increases since several systems have to be trained.

7.9 Comparison

In section 1.3 several other approaches to the word alignment task have been presented. All these authors have also done experiments on the English-French task. They are not all comparable, since they sometimes use additional data, but some comparisons can be made.
Moore and Takar/Lacoste-Julien mainly use the dice-coefficient as the only knowledge source for their word alignment system. This task is perhaps the most similar to the systems that only use the IBM1-lexica. But probably the IBM1-lexica already give more information than the Dice-coefficient. Therefore, no comparison was made.
In their best results they additionally used the IBM4-links and some HMM-links generated by a different generative approach. Only Blunsom/Cohn and Taskar/Locaste-Julien also give some results where they only use the IBM4-links. So these systems should be comparable with the best results of this work. It also has to be mentioned, that Taskar/Locaste-Julien do not use the GIZA-fertilities or the IBM4-lexica. But since the links are a very important feature, this should be comparable.
Blunsom/Cohn had an alignment error rate of 5.29 in their best system. So this result is 1 AER worse than the best result obtained by the discriminative word alignment presented in this work.
Taskar/Lacoste-Julien could reach an alignment error rate of 4.5 by using the IBM4 links in their second approach. This is still a little bit worse than the best results

Table 7.12: Comparison to other approaches

Name	Test
Blunsom/Cohn	5.29
Taskar/Lacoste-Julien	4.5
Taskar/Lacoste-Julien + HMM	3.8
Moore (incl. HMM)	3.7
CRF	4.30

obtained by this word alignment. But as already mentioned before they used less information.

Moore et al. only presented some results where they also used the links generated by another generative approach. There they presented an AER of 3.7. Taskar/Lacoste-Julien could improve from 4.5 to 3.8 by including this feature, so perhaps their approach has got a similar quality. But they could already reach an alignment error rate of 4.9 by only using the dice coefficient as knowledge source.

All in all the discriminative word alignment seems to be at least as good as the best approach presented in recent years.

8. Translation results

As already mentioned in the introduction machine translation is the main application for the word alignment. So the goal of this work was not only to generate better word alignments, but also to improve the translation quality of a SMT system. The experiments presented in this chapter will show the influence of different word alignment systems on the translation quality. Therefore, translation systems for different language pairs were build and evaluated.
First, the data used in these translation systems will be described in the next section. Afterwards, in section 8.2, small systems build to translate Spanish sentences into English will be analyzed. For this task the possibility to combine different word alignments was evaluated, too, and the influence of the word alignment in a reordering approach presented in [RoVo07] was explored. In the next section a bigger translation system for the inverse direction was evaluated. At last, the performance of a Chinese-English translation system using the discriminative word alignment was evaluated.

8.1 Data

The word alignments were tested on three different translation tasks. First, a small Spanish-English system was build. Furthermore, the translation quality was evaluated on a bigger English-Spanish system. At last, a system to translate Chinese sentences into English was trained and evaluated. In this part, the data used in the translation system will be described.

8.1.1 Small Spanish-English system

The translation system was trained on a training corpus of 100,000 sentences. These sentences were randomly selected from the the Final Edition of the European Parliament Proceedings provided for the TC-Start 07 Evaluation, which consists of 1.4 million sentences. Only sentences up to 80 words and a maximal fertility of 8 were selected.
The scaling factors of the decoder were optimized on the development set of the TC-Star 07 Evaluation and the results were reported on the test set of this evaluation.

Both sets contain some in-domain data from the European parliament and some out-of-domain data from the CORTES-corpus. The development set contained 1452 sentences and the test set 1470 sentences.

8.1.2 English-Spanish system

The English-Spanish translation system was trained on the whole Final Edition of the European Parliament Proceedings provided for the TC-Start 07 Evaluation. Here, sentences that were longer than 100 words were just cut off. No additional sentence selection has been applied.
The test set for this task contains no out-of-domain data and was also provided for the TC-Star 07 evaluation. The development set contains 1122 sentences of speeches in the European parliament and the test set consists out of 1130 sentences.

8.1.3 Chinese-English system

The translation system for the Chinese-English task uses the FBIS corpus. This corpus consists of 341K sentences, which contain 9M Chinese words and 11.5M English words. The Chinese text was segmented with the Stanford word segmenter like it was done with the hand-aligned data in the last section. The English text is lower cased.
The translations were evaluated on the MT'06 eval set. Before, the MT'03 eval set was used as development data. The test set consists of 1664 sentences and the development set of 919 sentences.

8.2 Small Spanish-English System

To analysis the influence of the word alignment on the translation quality, different word alignment systems were build. First, their word alignment performance was tested. This was done on the the same hand-aligned data than the English-Spanish system in section 7.4. Afterwards the systems were used to align the training corpus for the translation system and then, using this alignment, the phrase table was extracted. Only a subset of the original corpus was used as training corpus for the MT system to be able to test different word alignments.
The baseline system uses both GIZA++-alignments and combines them using the grow-diag-final heuristic. This heuristic was selected since it is the default one in the Pharaoh-Toolkit. The GIZA++-Toolkit was trained on the whole EPPS-corpus. All other systems use alignments generated by the discriminative word alignment. The features used in these systems are the same like the "link feature" system used in the last section. They all differ only in the way they are trained. The first system, "ML", was optimized by maximizing the log-likelihood of the correct alignment. The other systems were trained towards the AER and the F-score respectively. For the systems optimized towards the F-score different α-values were used. Consequently, the F(0.1) system is more trained towards recall and the F(0.9) system more towards precision as it can be seen in equation 2.13.
The word alignment results are shown in table 8.1. Every row shows the results for one system in different metrics. They are evaluated in different metrics to investigate

the correlation between word alignment and translation quality metrics. Therefore, the alignment error rate (AER), the F-score with different α-values between 0 to 1 and the consistent phrase error rate(CPER) were used. The F-score was calculated with sure and possible links as described in section 2.6. The F-score for $\alpha = 0$ equals the recall and the F-score for $\alpha = 1$ equals the precision. The CPER was extended similar to the F-score to be able to handle also sure and possible links. Then the CPER is defined as:

$$CPER(G,S,P) = 1 - \frac{2 * PhraseRecall(G,A) * PhrasePrecision(G,A)}{PhraseRecall(G,A) + PhrasePrecision(G,A)} \quad (8.1)$$

with S the set of sure links, P the set of possible links and *PhraseRecall* and *PhrasePrecision* defined like in equation 2.16.

First, these results show that the optimization of the discriminative word alignment works. So every system performs best in the metric it is optimized to. Furthermore in all metrics the discriminative word alignment could outperform the baseline system. The F(0.2) system is even better than the baseline system in all used metrics.

As mentioned before these systems were used to align the training corpus. After that, a phrase table is extracted using the Pharaoh-Toolkit. Then the STTK-Decoder is used to optimize its scaling factors towards BLEU and the test set is translated to measure the translation performance. The language model used in the decoder is a tri-gram model trained on the target part of the training corpus.

The results of the different translation systems are shown in table 8.2. The translations are evaluated with the NIST and BLEU metric. The results are shown for the whole test data as well as only the in-domain and only the out-of-domain data. The results for one system are shown in one line. For the results of the whole test set also the significance of the results were analyzed. All scores mark with a "*" are significantly better than the baseline system on a significance level of 5%.

The best translations according to the NIST and BLEU metric were generated with the systems trained towards AER and towards F(0.3). These systems could gain 0.74 BLEU points or 0.13 and 0.11 NIST points respectively compared to the baseline system on the test set. This is an significant improvement on a significance level of 5%. Furthermore, all systems optimized towards AER or the F-score except the one optimized towards an α-value of 0.8 outperform the baseline system and most of them are even significantly better. If the results on the test set are compared with the results on the development set, it is surprising that the discriminative word alignment systems could often gain that much on the test set although they are as good as the baseline system on the development set or even worse than the baseline system. One reason could be, that the translation systems using the discriminative word alignment have less problems with overfitting. This could be the case since the word alignment quality is better and so the extracted phrase pairs are better.

Recently, several experiments analyze the correlation between word alignment metrics and the translation quality. In this experiments it was also evaluated whether it is better to have a precision- or recall-orientate alignment. Therefore the Pearson correlation coefficient r^2 was calculated for different pairs of translation quality metric and a word alignment metric. The results are shown in table 8.3. In that table a row corresponds to a word alignment metric and a column to the translation metric on the shown test set. Again, the results for the translation metric were shown for the whole test set as well as for the in-domain and out-of-domain part.

Regarding the correlation coefficients it can be seen, that the performance on the

Table 8.1: Translation results for ES-EN

System	AER	F(0.0)	F(0.1)	F(0.2)	F(0.3)	F(0.4)	F(0.5)	F(0.6)	F(0.7)	F(0.8)	F(0.9)	F(1.0)	CPER
Baseline	19.26	80.67	80.68	80.70	80.71	80.72	80.74	80.75	80.77	80.78	80.79	80.81	30.78
ML	16.90	74.03	75.62	77.28	79.02	80.84	82.74	84.74	86.83	89.03	91.35	93.79	33.44
AER	**15.31**	78.13	79.30	80.50	81.75	83.03	84.35	85.71	87.12	88.57	90.08	91.63	27.47
F(0.1)	23.44	**84.58**	**82.98**	81.45	79.97	78.54	77.16	75.83	74.55	73.31	72.11	70.94	36.31
F(0.2)	17.21	81.61	81.83	**82.05**	82.27	82.49	82.72	82.94	83.17	83.40	83.63	83.86	28.21
F(0.3)	16.22	80.50	81.10	81.71	**82.32**	82.95	83.58	84.23	84.88	85.54	86.22	86.90	**26.88**
F(0.4)	15.66	79.51	80.38	81.27	82.18	**83.11**	84.06	85.04	86.03	87.05	88.10	89.17	26.93
F(0.5)	15.39	78.51	79.60	80.72	81.87	83.05	**84.27**	85.53	86.82	88.16	89.53	90.95	27.31
F(0.6)	15.88	75.60	77.10	78.66	80.29	81.98	83.75	**85.59**	87.52	89.53	91.64	93.86	28.90
F(0.7)	16.67	73.31	75.05	76.88	78.79	80.81	82.93	85.17	**87.53**	90.03	92.67	95.47	32.47
F(0.8)	17.89	70.82	72.76	74.81	76.98	79.28	81.72	84.32	87.09	**90.04**	93.20	96.60	36.60
F(0.9)	19.42	68.26	70.36	72.59	74.97	77.51	80.23	83.15	86.29	89.67	**93.33**	**97.30**	40.96

Table 8.2: Translation results for ES-EN

	Dev	Test-All		Test-In-Domain		Test-Out-of-Domain	
Name	BLEU	BLEU	NIST	BLEU	NIST	BLEU	NIST
GIZA	**40.07**	40.56	9.48	43.87	9.48	37.16	8.70
ML	39.36	40.22	9.43	43.12	9.40	37.27	8.68
AER	39.98	**41.30***	**9.61***	44.36	**9.56**	38.18	8.84
F(0.1)	39.00	39.90	9.37	42.79	9.32	36.94	8.64
F(0.2)	40.07	40.87*	9.52*	44.08	9.51	37.57	8.73
F(0.3)	40.14	**41.30***	9.59*	**44.47**	**9.56**	38.05	8.82
F(0.4)	39.99	41.17*	9.60*	44.26	**9.56**	38.02	8.83
F(0.5)	39.88	40.99*	9.57*	44.11	9.54	37.79	8.79
F(0.6)	39.94	40.71	9.50	43.74	9.47	37.61	8.74
F(0.7)	39.76	41.01*	9.58*	44.06	**9.56**	37.92	8.80
F(0.8)	39.26	40.25	9.45	43.35	9.43	37.07	8.68
F(0.9)	39.14	41.07*	9.60*	43.84	9.52	**38.24**	**8.87**

Table 8.3: Correlation between word alignment and translation quality metric

	Dev	Test-All		Test-In-Domain		Test-Out-of-Domain	
WA-Metric	BLEU	BLEU	NIST	BLEU	NIST	BLEU	NIST
AER	0.47	0.46	0.43	0.50	0.54	0.33	0.28
F(0.0)	0.16	0.00	0.04	0.01	0.02	0.06	0.08
F(0.1)	0.24	0.00	0.02	0.03	0.02	0.03	0.04
F(0.2)	0.37	0.02	0.00	0.08	0.01	0.01	0.01
F(0.3)	0.60	0.11	0.03	0.23	0.09	0.01	0.00
F(0.4)	0.76	0.33	0.22	0.48	0.35	0.14	0.10
F(0.5)	0.52	0.46	0.42	0.52	0.53	0.32	0.27
F(0.6)	0.22	0.39	0.42	0.35	0.47	0.35	0.32
F(0.7)	0.08	0.29	0.36	0.23	0.37	0.32	0.30
F(0.8)	0.02	0.23	0.30	0.15	0.30	0.28	0.28
F(0.9)	0.00	0.18	0.26	0.11	0.24	0.25	0.25
F(1.0)	0.00	0.14	0.23	0.08	0.20	0.23	0.23
CPER	0.82	0.27	0.16	0.44	0.28	0.09	0.06

development data could be predicted quite good using the word alignment metrics. The best correlation was reached using the CPER. Furthermore, the F-score with an α-value of 0.4 or 0.3 predicts the translation quality quite well. This indicated that a recall-oriented alignment performs better on the development set as it was reported in [FrMa07]. This assumption is assisted by the results shown in table 8.2. On the development set the systems optimized towards a F-score with a lower α-value performs better than the one with a larger value. These are the systems that are more concentrating on the recall than on the precision.

On the test set the picture is more complicated. Here, no word alignment metric could predict the translation quality well. In contrast to other authors the AER performs best compared to all other word alignment metrics. It is not surprising that here the prediction is more complicated since precision-oriented as well as recall-oriented systems generate good and bad translations of the test set. Furthermore, the system performance on the test set can even not be predicted well by looking at the translation quality of the development set. Consequently, it is good to analyze the in-domain and out-of-domain part of the test set separately as proposed in [WuWa07].

First, on the in-domain part, the recall-orientated systems generate better translations than the precision-orientated ones. All word alignments optimized to a F-score between 0.2 and 0.5 get a higher BLUE score on the in-domain part than all systems optimized towards a F-score with an α-value bigger than 0.5. Furthermore, the F-score with an α-value lower than 0.5 seems to have a sightly better correlation with translations metrics than the more precision-oriented F-scores.

On the out-of-domain part the picture is different. Here, the best system is the F(0.9) system and some other precision-oriented word alignments perform as good as the recall oriented ones. Here the correlation between the word alignment metrics and translation metrics is even worse, but the precision-oriented ones perform a little bit better. So the precision seems to be more important for the out-of-domain data as for the in-domain data. These results seem to be consistent with the results reported in [WuWa07]

8.2.1 System combination

In [WuWa07] Wu and Wang reported some improvements by combining different word alignments. When looking at the last results, the most promising systems to be combined seem to be the F(0.3) and F(0.9) systems. On the one hand, both system have got a quite good performance on the whole test set. On the other hand, both systems are very different. The F(0.3) system is more recall-oriented and performs best on the in-domain data, but does not generate that good translations for the out-of-domain data. In contrast, the F(0.9) system is concentrating more on the precision and leads to better results on the out-of-domain data. Both systems are combined using two different methods.

In the first method the phrase extraction is done on the concatenation of both aligned copra. Then the relative frequencies used in the phrase table are calculated with both alignments. This method is similar to the one described as "count merging" by Wu et al. Once a simple concatenation of both corpora was done (System count1:1) and once the copra aligned by the system F(0.3) is used twice (System count2:1). In the second case the links of the F(0.3) where weighted double compared to the other

Table 8.4: Translation results for ES-EN System combination

	Dev	Test-All		Test-In-Domain		Test-Out-of-Domain	
Name	BLEU	BLEU	NIST	BLEU	NIST	BLEU	NIST
F(0.3)	**40.14**	**41.30**	9.59	**44.47**	**9.56**	38.05	8.82
F(0.9)	39.14	41.07	**9.60**	43.84	9.52	**38.24**	**8.87**
Count1:1	39.70	41.22	9.57	44.21	9.53	38.17	8.81
Count2:1	40.12	41.18	9.57	44.27	9.54	38.02	8.79
Model	39.60	40.42	9.45	43.69	9.43	37.05	8.67

system. The F(0.3) system was weighted double, since it performance is better than the one of the other system.

In the other method, the phrase tables of both systems were merged. Therefore, the scores of the phrases were simply concatenated and two extra features were added to indicate in which phrase table the phrase pair exists. Here the weights between both alignment methods can be optimized during the optimization of the translation system.

The results are shown in table 8.4. The system "count1:1" could perform better than F(0.9) on the in-domain data and better than F(0.3) on the out-of-domain data and on both tasks it is worse than the other system. So the "count1:1" is a more general system, but its performance on the whole test set is not better than the F(0.3) system. As expected, the system count2:1 does behave more like the F(0.3) system than the count1:1 system, but this does not improve the overall performance. In contrast to Wu and Wang, this method neither improves the translation quality on the out-of-domain data nor on the in-domain data.

The second approach does even perform worse than the first approach. The results on all tasks are worse than the ones of the F(0.3) system and the F(0.9) system. So these methods do not seem to be a good way to combine different alignments.

8.2.2 Lattice reordering

The word alignment is also an knowledge source for the lattice reordering model presented in [RoVo07]. Therefore, in this section the effect of the discriminative word alignment models on this model is tested. The rules for the lattice reordering were extracted from the small Spanish-English corpus used for the experiments described before. Then the rules were used to build lattices for the development and test set. The lattices were translated using the "AER" System described before.

The Baseline system uses the alignment generated by the grow-diag-final heuristic. The other system are the same discriminant word alignments as before optimized towards different metrics.

The results for these experiments are shown in table 8.5. The results which are significantly better than the baseline system at a level of 5%, are marked by a *. This time only the results for the whole test set are shown because there was no different behavior on the different parts.

The best results on the test set were obtained by the word alignment optimized towards a F-score with $\alpha = 0.3$ and $\alpha = 0.8$. The first system could gain 0.27 BLEU points and 0.07 NIST points and its NIST score is significantly better. The second

Table 8.5: Translation results for Lattice Reordering

	Dev	Test-All	
Name	BLEU	BLEU	NIST
Baseline	40.47	42.48	9.70
AER	40.77	42.36	9.68
F0.1	40.78	42.47	9.69
F0.2	40.74	42.39	9.58
F0.3	40.78	42.75	**9.77***
F0.4	40.75	42.50	9.72
F0.5	40.80	42.38	9.68
F0.6	**40.92**	42.55	9.70
F0.7	40.82	42.04	9.62
F0.8	40.90	**42.85***	9.73*
F0.9	40.90	42.51	9.68

system could improve the BLEU score by 0.37 points and the NIST score by 0.03 points. This time both scores are improved significantly. For the best systems the gain on the development set is nearly the same like the one on the development set.

8.2.3 Conclusion

The new discriminative word alignment approach could improve the translation quality significantly. If the word alignment is only used for the phrase table, the BLEU score could be improved by 0.74 points to 41.30 points. Furthermore, the translation quality could be improved, if the new word alignment is used for a reordering model. This leaded to an improvement of 0.37 to 42.85 BLEU points. In an additional experiment the baseline word alignment system was used for the phrase table as well as for the reordering model. This system reached a BLEU score of 42.00 and a NIST score of 9.63. The best system using discriminative word alignments use the word alignment model optimized towards AER for the phrase table and the word alignment optimized towards an F-score with an α-value of 0.8 for the reordering model. If these systems are compared the discriminative word alignment could gain 0.85 BLEU points, if it is used for both models.

8.3 English-Spanish system

After having analyzed the influence of the new word alignment on a small system in the last section, in this section the influence of the word alignment on a larger system is evaluated. Therefore, two large English-Spanish systems were build. The systems use different word alignments, but the remaining system is the same. The phrase table was built by using the Pharaoh-Toolkit. The language model was trained on the whole corpus and a reordering window of 2 was used. Then the systems were optimized towards BLEU and evaluated on the test set.
The Baseline system uses the alignments generated by the GIZA++Toolkit for both directions and combines them using the grow-diag-final heuristic. The GIZA++Toolkit is configured like proposed in the scripts of the Pharaoh-Toolkit.

Table 8.6: Translation results for EN-ES

	Dev	Test-All	
Name	BLEU	NIST	BLEU
Baseline	40.04	9.74	47.73
DWA	**41.62**	**9.81***	**48.13**

Table 8.7: Translation results for CH-EN

	Dev	Test-All	
Name	BLEU	BLEU	NIST
Baseline	27.13	22.56	8.45
AER	**27.63**	23.85*	**8.48**
F0.3	26.34	22.35	8.24
F0.7	26.40	23.52*	8.39
Phrase feature AER	25.84	23.42*	8.44
Phrase feature F0.7	26.41	**23.92***	8.46

The discriminate word alignment (DWA) is trained and tested on the same hand-aligned English-Spanish data used in the experiments before. The system is trained towards AER and uses the same features like the "link feature" system in the last section. The results of the word alignment were already discussed in the last section and the discriminative word alignment could improve the AER by 3.95 points.

The results are shown in table 8.6. The discriminative word alignment outperforms the baseline system by 0.4 BLEU points and 0.07 NIST points. The NIST score has significantly improved at a significance level of 5%. The improvement is around half as much as in the small system. One reason could be, that the quality of the word alignment gets less important in bigger system, because some mistakes are not that important if there are enough examples.

8.4 Chinese-English system

In addition to the translation tasks of European languages in the last sections, systems translating Chinese to English were evaluated. Therefore, the training corpus was aligned with different methods. Afterwards, the phrase pairs were extracted with the Pharaoh-Toolkit. Then this phrase table and a tri-gram language model trained on 100M English words were used to translate the development and test set using a reordering window of 3. The system was optimized towards BLEU for 8 iterations.

The results on the development and test set are shown in table 8.7. The translations were evaluated using the BLEU and NIST metric. The scores marked with an * are significantly better than the baseline system at a significance level of 5%.

For the baseline system the GIZA++-Toolkit generates a word alignment for every direction. Afterwards these alignments were combined using the grow-diag-final heuristics. Using this alignment the translation system reached a BLEU-score of 22.56 and a NIST-score of 8.45.

The other systems use discriminative word alignments. The first three systems use the same features like the "add. Directions" system in section 7.6. They differ in the way they are trained. The first system is optimized towards the AER and the other two towards the F-score with an α-value of 0.3 and 0.7 respectively. The other two systems use in addition the phrase features. The first one is trained towards the AER, but uses the phrases of the F0.7 system, since this leads to better results of the word alignment measured in the AER than the phrases of the AER system. The second is trained towards the F-score with an α-value of 0.7 and uses the phrases generated by the AER. Again, this leads to better results on the word alignment test set measure in the F-score with an α-value of 0.7 than using the phrases generated by the F0.7 system.

In the BLEU metric the systems trained towards the AER and an F-score with an α-value of 0.7 are significantly better than the baseline system. The NIST-scores of these systems are comparable to the one of the baseline system. The system trained towards F0.3 performs worse than the baseline system in both metrics. This indicates that the precision is more important in this task than in the European language tasks. There the system trained towards an F-score with an α-value of 0.3 performed quite good.

The systems using the additional phrase features could both improve the word alignment quality. In the translation quality this is only the case for the system trained towards the F-score. Then this system achieves the best BLEU-score. The system trained towards the AER performs worse than the one without the phrase features. Another important point in this experiments is the difference in the performance on the development and test set. The systems using the discriminative word alignments perform often worse than the baseline system on the development set, but could outperform it on the test set significantly.

9. Conclusion

In this work new discriminative word alignment models were presented. The described models differ in the used knowledge sources, the optimization methods and the use of regularization techniques. They use a conditional random field to model directly the word alignment matrix. This enables the models to generate every possible alignment. Furthermore, since these models are symmetric, no heuristics to combine the alignments of both directions are needed. The models have to be trained on a small hand-aligned development set, but already 100 sentences of development data are enough to generate good word alignments. Furthermore, since the training can be done quite fast, the alignment can be done independently for every sentence. So the alignment can be done in parallel, which is important for the use of growing corpora.

In contrast to generative models, like the common used GIZA++-Toolkit, it is possible to integrate easily all available information into these models. So POS-tags or the similarity of words can be used as an additional hint which words should be aligned. It is shown in the experiments that the performance can be increased by using additional information.

Since the CRFs are quite complex the features have to fulfil some constraints to be able to do the calculation efficiently. So only fertilities up to a given maximal fertility are considered. But the experiments show, that this does not seem to hurt the system performance. Furthermore, dynamic programming was used to make the calculations efficiently.

Different methods to train the models were presented. Besides form the standard approach to maximize the log-likelihood, the models can be trained towards an approximation of word alignment metrics. This was done for the alignment error rate and the F-score. Using the optimization towards the F-score it is possible to generate alignments more concentration on the precision or on the recall.

The presented word alignment approach uses CRFs, a model that has already been used in many other applications. This enables the models to profit from improvements made in these areas. For example, an approach to prevent overfitting was presented, which uses the LOP-CRF framework used before for the Named Entity Recognition. This could improve systems in some cases, but has got the disadvantage, that additional training data is needed.

The experiments show, that these models could improve the word alignment quality compared to the GIZA++-Toolkit. In the English-Spanish task the alignment error rate could be decreased by 1.71 points or 10% compared to the best combination of both GIZA++-alignments. In the English-French systems the alignment error rate could be improved by 1.03 points or 20%. The results on the French-English task are not completely comparable to other publications, but the best model seems to be at least as good as the best systems published before.

Furthermore the quality of the translation systems using the alignment was evaluated. Here, the new alignment models could significantly improve the system. The performance could be improved by using the word alignment for the phrase extraction as well as using the word alignment to learn reordering rules. On the Spanish-English task the performance could be improved by 0.85 BLEU points and on the Chinese-English task even by 1.36 BLEU points. Furthermore, it was shown, that the possibility to generate alignments focusing more on recall or precision could help to improve the translation quality.

9.1 Further work

Although the models use already many knowledge sources, there are some left, that could help to improve the word alignment quality even more. For example, syntax tree information could be used to improve the translation quality. Furthermore, language specific features could be used for difficult constructions. For example, in a German-English task, the dependency between parts of the verb that are disjoint could be modeled. Of course, if would be interesting in general to see the performance on other language pairs.

Furthermore, the problem of overfitting could be investigated in more detail. The LOP-CRF does only improve the quality of some systems and often they are improved a little bit, but in some systems the overfitting is a bigger problem. Furthermore, this gets more important if even more features are added.

The possibility to optimize towards different metrics helped a lot, when looking at the translation quality. Only with the F-score optimization the quality of the lattice reordering could be improved. Here, the consistent phrase error rate (CPER) would be an interesting metric. Especially, as it already incorporated information of the phase extraction.

Literatur

[AyDo06] Necip Fazil Ayan und Bonnie J. Dorr. Going beyond AER: an extensive analysis of word alignments and their impact on MT. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, Morristown, NJ, USA, 2006. Association for Computational Linguistics, S. 9–16.

[BlCo06] Phil Blunsom und Trevor Cohn. Discriminative word alignment with conditional random fields. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, Morristown, NJ, USA, 2006. Association for Computational Linguistics, S. 65–72.

[BPPM94] Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra und Robert L. Mercer. The Mathematic of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2), 1994, S. 263–311.

[BTKo02] Pieter Abbeel Ben Taskar und Daphne Koller. Discriminative Probabilistic Models for Relation Data. In *In Eighteenth Conference of Uncertainty in Artificial Intelligence*, 2002.

[FrMa07] Alexander Fraser und Daniel Marcu. Measuring Word Alignment Quality for Statistical Machine Translation. *Computational Linguistics*, 33(3), 2007, S. 293–303.

[GWLC06] Sheng Gao, Wen Wu, Chin-Hui Lee und Tat-Seng Chua. A maximal figure-of-merit (MFoM)-learning approach to robust classifier design for text categorization. *ACM Trans. Inf. Syst.*, 24(2), 2006, S. 190–218.

[Hesk98] Tom Heskes. Selecting weighting factors in logarithmic opinion pools. In *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, Cambridge, MA, USA, 1998. MIT Press, S. 266–272.

[KoOM03] Philipp Koehn, Franz Josef Och und Daniel Marcu. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, Morristown, NJ, USA, 2003. Association for Computational Linguistics, S. 48–54.

[LaMP01] John Lafferty, Andrew McCallum und Fernando Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proc. 18th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 2001, S. 282–289.

[LJTKJ06] Simon Lacoste-Julien, Ben Taskar, Dan Klein und Michael I. Jordan. Word alignment via quadratic assignment. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Morristown, NJ, USA, 2006. Association for Computational Linguistics, S. 112–119.

[LRHB06] Xiangyang Lan, Stefan Roth, Daniel P. Huttenlocher und Michael J. Black. Efficient Belief Propagation with Learned Higher-Order Markov Random Fields. In Ales Leonardis, Horst Bischof und Axel Pinz (Hrsg.), *ECCV (2)*, Band 3952 der *Lecture Notes in Computer Science*. Springer, 2006, S. 269–282.

[McLi03] A. McCallum und W. Li. Early results for named entity recognition with conditional random fields, 2003.

[Moor05] Robert C. Moore. A discriminative framework for bilingual word alignment. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Morristown, NJ, USA, 2005. Association for Computational Linguistics, S. 81–88.

[MoYB06] Robert C. Moore, Wen tau Yih und Andreas Bode. Improved discriminative bilingual word alignment. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, Morristown, NJ, USA, 2006. Association for Computational Linguistics, S. 513–520.

[OcNe00] F. Och und H. Ney. Improved statistical alignment models, 2000. To appear.

[OcNe03] Franz Josef Och und Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1), 2003, S. 19–51.

[Pear88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 1988.

[PLbM05] Rafael Banchs Patric Lambert, Adria de Gispert und Jose b. Marino. Guidelines for Word Alignment Evaluation and Manual Alignment. *Language Resources and Evaluation*, 2005, S. 267–285.

[RoVo07] Kay Rottman und Stephan Vogel. Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model. In *TMI '07*, 2007.

[ShPe03] F. Sha und F. Pereira. Shallow parsing with conditional random fields, 2003.

[SmCO05] Andrew Smith, Trevor Cohn und Miles Osborne. Logarithmic opinion pools for conditional random fields. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, Morristown, NJ, USA, 2005. Association for Computational Linguistics, S. 18–25.

[SmOs05] Andrew Smith und Miles Osborne. Regularisation Techniques for Conditional Random Fields: Parameterised Versus Parameter-Free. In *Lecutre Notes in Computer Science , Natural Language Processing - IJCNLP 2005, Volume 3651/2005*. Springer Berlin / Heidelberg, 2005, S. 896–907.

[SuMI06] Jun Suzuki, Erik McDermott und Hideki Isozaki. Training conditional random fields with multivariate evaluation measures. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, Morristown, NJ, USA, 2006. Association for Computational Linguistics, S. 217–224.

[TaAK02] Benjamin Taskar, Pieter Abbeel und Daphne Koller. Discriminative Probabilistic Models for Relational Data. In Adnan Darwiche und Nir Friedman (Hrsg.), *UAI*. Morgan Kaufmann, 2002, S. 485–492.

[TaLJK05] Ben Taskar, Simon Lacoste-Julien und Dan Klein. A discriminative matching approach to word alignment. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Morristown, NJ, USA, 2005. Association for Computational Linguistics, S. 73–80.

[ViPN06] David Vilar, Maja Popovic und Hermann Ney. AER: Do we need to "improve" our alignments? In *Proc. of the International Workshop on Spoken Language Translation*, Kyoto, Japan, 2006. S. 205–212.

[VoNT96] Stephan Vogel, Hermann Ney und Christoph Tillmann. HMM-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics*, Morristown, NJ, USA, 1996. Association for Computational Linguistics, S. 836–841.

[WaWa98] Y. Wang und A. Waibel. Fast decoding for statistical machine translation, 1998.

[WuWa07] Hua Wu und Haifeng Wang. Comparative Study of Word Alignment Heuristics and Phrase-Based SMT. In *MT Summit '07*, 2007.

[YaKn01] Kenji Yamada und Kevin Knight. A Syntax-based Statistical Translation Model. In *Meeting of the Association for Computational Linguistics*, 2001, S. 523–530.

[YeFW03] Jonathan S. Yedidia, William T. Freeman und Yair Weiss. Understanding belief propagation and its generalizations. 2003, S. 239–269.