Daiana Azamat

Statistical Morphological Disambiguation for Kazakh Language

Mathematics Major

Capstone project

Advisor:  
Zh. Assylbekov, PhD

Second reader:  
A. Makazhanov, MSc

Astana — 2016
Content

1 Introduction ................................................................. 3

2 Kazakh Language ......................................................... 4

3 Statistical morphological disambiguation .......................... 6
   3.1 Derivation ............................................................. 7
   3.2 Parameters estimation ............................................. 9
   3.3 Tagging with the Viterbi algorithm .............................. 12

4 Conclusion ................................................................. 13
Abstract

This paper presents the results of developing a statistical model for morphological disambiguation of Kazakh text. Starting with basic assumptions we tried to cope with the complex morphology of Kazakh language by breaking up lexical forms across their derivational boundaries into inflectional groups and modeling their behavior with statistical methods. We also provide maximum likelihood estimates for the parameters and an effective way to perform disambiguation with the Viterbi algorithm.
Chapter 1

Introduction

In this paper, we present a statistical model for morphological disambiguation of Kazakh text. Morphological disambiguation is the task of selecting the sequence of morphological parses corresponding to a sequence of words, from the set of possible parses for those words. Morphological disambiguation is an important step for a number of natural language processing (NLP) tasks and this importance becomes more crucial for agglutinative languages such as Kazakh, Turkish, Finnish, Hungarian, etc.

Kazakh language (as well as any morphologically rich language) presents an interesting problem for statistical natural language processing since the number of possible morphological parses is very large due to the productive derivational morphology [1, 15]. Morphological disambiguation of inflectional and agglutinative languages was inspired by part-of-speech (POS) tagging techniques. It involves determining not only the major or minor parts-of-speech, but also all relevant lexical and morphological features of forms. Previous approaches to morphological disambiguation of Turkish text had employed constraint-based methods (Oflazer and Kuruöz [18]; Oflazer and Tür [19, 20]), statistical methods (Hakkani-Tür et al. [11], or both (Yuret and Türe [22], Kutlu and Cicekli[13]). Based on the approach by Hakkani-Tür et al., this work describes statistical morphological disambiguation model for Kazakh language.

In Chapter 2, relevant properties of Kazakh language are presented. In Chapter 3, first the statistical model for morphological disambiguation is described, and then maximum likelihood estimates (MLE) for the parameters of this model are provided. Some discussion of drawbacks of MLE is given. We finally show how to perform disambiguation effectively with the Viterbi algorithm.
Kazakh Language

Kazakh (natively қазақ тілі, қазақша) is a Turkic language belonging to the Kypchak (or Qıpçaq) branch, closely related to Nogay (or Noğay) and Qaraqalpaq. It is spoken by around 13 million people in Kazakhstan, China, Mongolia, and adjacent areas [14].

Kazakh is an agglutinative language, which means that words are formed by joining suffixes to the stem. A Kazakh word can thus correspond to English phrases of various length as shown below:

дос friend
dостар friends
dостарым my friends
dостарымыз our friends
dостарымыздамыз at our friends
dостарымыздамыз мыз we are at our friends

The effect of rich morphology can be observed in parallel Kazakh-English texts. Table below provides the vocabulary sizes, type-token ratios (TTR) and out-of-vocabulary (OOV) rates of Kazakh and English sides of a parallel corpus used in [2].

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>Kazakh</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vocabulary size</td>
<td>18,170</td>
<td>35,984</td>
</tr>
<tr>
<td>Type-token ratio</td>
<td>3.8%</td>
<td>9.8%</td>
</tr>
<tr>
<td>OOV rate</td>
<td>1.9%</td>
<td>5.0%</td>
</tr>
</tbody>
</table>

It is easy to see that rich morphology leads to sparse data problems for statistical natural language processing of Kazakh, be it tasks in machine translation, text categorization, sentiment analysis, etc. A common approach (see [10, 4, 17, 3]) applied for morphologically rich languages is to convert surface forms into lexical forms (i.e. analyze words), and then perform some morphological segmentation.
for the lexical forms (i.e. split analyzes). The segmentation schemes are usually motivated by linguistics and the domain of intended use. For example, for a Kazakh-English word alignment task we could be in favor of the following segmentation of the above mentioned word достарымыздамыз:

<table>
<thead>
<tr>
<th>дос</th>
<th>⟨n⟩⟨pl⟩</th>
<th>друзья</th>
<th>⟨px1pl⟩</th>
<th>да</th>
<th>+э⟨cop⟩</th>
<th>⟨p1⟩⟨pl⟩</th>
</tr>
</thead>
<tbody>
<tr>
<td>friends</td>
<td>our</td>
<td>at</td>
<td>are</td>
<td>we</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

since each segment of the Kazakh word would then correspond to a single word in English. The problem is that often for a word in Kazakh we have more than one way to analyze it, as in the example below:

‘in 2009, we started the construction works.’

2009 жылы біз құрылысты бастадық.

жылы⟨adj⟩ ‘warm’

жылы⟨adj⟩⟨advl⟩ ‘warmly’

→ жыл⟨n⟩⟨px3sp⟩⟨nom⟩‘year’

жылы⟨adj⟩⟨subst⟩⟨nom⟩‘warmth’

Selecting the correct analysis from among all possible analyses is called morphological disambiguation. Due to productive derivational morphology this task itself suffers from data sparseness. To alleviate the data sparseness problem we break down the full analyses into smaller units — inflectional groups. An inflectional group is a tag sequence split by a derivation boundary. For example, in the sentence that follows, the word айналасындарға ‘to the ones in his vicinity’ is split into root \( r \) and two inflectional groups, \( g_1 \) and \( g_2 \), the first containing the tags before the derivation boundary -ғы and the second containing the derivation boundary and subsequent tags.

Жәңгір хан мен оның айналасындарға …

(айнала)⟨syn·da⟩(ғы·лар·ға)

(айнала)⟨n·px3sp·loc⟩⟨subst·pl·dat⟩

We will heavily exploit the following observation of dependency relationships which was made by Hakkani-Tür et al. [11, p. 387] for Turkish, but is valid for Kazakh as well: When a word is considered to be a sequence of inflectional groups, syntactic relation links only emanate from the last inflectional group of a (dependent) word, and land on one of the inflectional groups of the (head) word on the right.

\(^{1}\)hereinafter we use the Apertium tagset [7] for analyzed forms
Chapter 3

Statistical morphological disambiguation

Following [5], we will use the notation in Table 3.1. We use subscripts to refer to words and tags in particular positions of the sentences and corpora we tag. We use superscripts to refer to word types in the lexicon of words and to refer to tag types in the tag set.

The basic mathematical object with which we deal here is the joint probability distribution \( \Pr(W = w, T = t) \), where the random variables \( W \) and \( T \) are a sequence of words and a sequence of tags. We also consider various marginal and conditional probability distributions that can be constructed from \( \Pr(W = w, T = t) \), especially the distribution \( \Pr(T = t) \). We generally follow the common convention of using uppercase letters to denote random variables and the corresponding lowercase letters to denote specific values that the random variables may take. When there is no possibility for confusion, we write \( \Pr(w, t) \), and use similar shorthands throughout.

In this compact notation, morphological disambiguation is the problem of selecting the sequence of morphological parses (including the root), \( t = t_1t_2 \cdots t_n \),

<table>
<thead>
<tr>
<th>Table 3.1: 'Notation'</th>
</tr>
</thead>
<tbody>
<tr>
<td>( w_i ) the word (token) at position ( i ) in the corpus</td>
</tr>
<tr>
<td>( t_i ) the tag of ( w_i )</td>
</tr>
<tr>
<td>( w_{i,i+m} ) the words occurring at positions ( i ) through ( i + m )</td>
</tr>
<tr>
<td>( t_{i,i+m} ) the tags ( t_i \cdots t_{i+m} ) for ( w_i \cdots w_{i+m} )</td>
</tr>
<tr>
<td>( r_i ) the root of ( w_i )</td>
</tr>
<tr>
<td>( g_{i,k} ) the ( k )-th inflectional group of ( w_i )</td>
</tr>
<tr>
<td>( n ) length of a text chunk</td>
</tr>
<tr>
<td>( \text{(be it a sentence, a paragraph or a whole text)} )</td>
</tr>
<tr>
<td>( w ) the words ( w_{1,n} ) of a text chunk</td>
</tr>
<tr>
<td>( t ) the tags ( t_{1,n} ) for ( w_{1,n} )</td>
</tr>
</tbody>
</table>
corresponding to a sequence of words $w = w_1w_2 \cdots w_n$, from the set of possible parses for these words:

$$\arg\max_t \Pr(t|w).$$

(3.1)

Using Bayes' rule and taking into account that $w$ is constant for all possible values $t$, we can rewrite (3.1) as:

$$\arg\max_t \frac{\Pr(t) \times \Pr(w|t)}{\Pr(w)} = \arg\max_t \Pr(t) \times \Pr(w|t)$$

(3.2)

In Kazakh, given a morphological analysis\(^1\) including the root, there is only one surface form that can correspond to it, that is, there is no morphological generation ambiguity. Therefore,

$$\Pr(w|t) = 1,$$

and the morphological disambiguation problem (3.2) is simplified to finding the most probable sequence of parses:

$$\arg\max_t \Pr(t)$$

(3.3)

Keep in mind that the search space in equations (3.1)–(3.3) is not equal to the set of all hypothetically possible sequences $t$. Instead it is limited to only the set of parse sequences that can correspond to $w$. Such limited set is obtained as a full or constrained output of a morphological analysis tool.

### 3.1 Derivation

Using the chain rule, the probability in (3.3) can always be rewritten as:

$$\Pr(t) = \prod_{i=1}^n \Pr(t_i|t_{1,i-1}).$$

(3.4)

It is important to realize that equation (3.4) is not an approximation. We are simply asserting in this equation that when we generate a sequence of parses, we can firstly choose the first analysis. Then we can choose the second parse given our knowledge of the first parse. Then we can select the third analysis given our knowledge of the first two parses, and so on. As we step through the sequence, at each point we make our next choice given our complete knowledge of the all our previous choices. The conditional probabilities on the right-hand side of equation (3.4) cannot all be taken as independent parameters because there are too many of them. In the bigram model, we assume that

$$\Pr(t_i|t_{1,i-1}) \approx \Pr(t_i|t_{i-1}).$$

\(^1\)We use the terms morphological analysis or parse interchangeably, to refer to individual distinct morphological parses of a token.
That is, we assume that the current analysis is only dependent on the previous one. With this assumption we get the following:

\[ \Pr(t) \approx \prod_{i=1}^{n} \Pr(t_i | t_{i-1}). \]  

(3.5)

However, the probabilities on the right-hand side of this equation still cannot be taken as parameters, since the number of possible analyzes is very large in morphologically rich languages. Following the discussion from Section 2 we split morphological parses across their derivational boundaries, i.e. we consider morphological analysis as a sequence of root \((r_i)\) and inflectional groups \((g_{i,k})\), and therefore, each parse \(t_i\) can be represented as \((r_i, g_{i,1}, \ldots, g_{i,n_i})\). Then the probabilities \(\Pr(t_i | t_{i-1})\) can be rewritten as:

\[
\Pr(t_i | t_{i-1}) \\
= \Pr((r_i, g_{i,1}, \ldots, g_{i,n_i}) | (r_{i-1}, g_{i-1,1}, \ldots, g_{i-1,n_{i-1}})) \\
= \{ \text{chain rule} \} = \Pr(r_i | (r_{i-1}, g_{i-1,1}, \ldots, g_{i-1,n_{i-1}})) \\
\times \Pr(g_{i,1} | (r_{i-1}, g_{i-1,1}, \ldots, g_{i-1,n_{i-1}}, r_i) \times \ldots \times \\
\times \Pr(g_{i,n_i} | (r_{i-1}, g_{i-1,1}, \ldots, g_{i-1,n_{i-1}}, r_i, g_{i,1}, \ldots, g_{i,n_i-1}) \\
(3.6)
\]

In order to simplify this representation we throw in the following independence assumptions

\[
\Pr(r_i | (r_{i-1}, g_{i-1,1}, \ldots, g_{i-1,n_{i-1}})) \approx \Pr(r_i | r_{i-1}), \\
(3.7)
\]

\[
\Pr(g_{i,k} | (r_{i-1}, g_{i-1,1}, \ldots, g_{i-1,n_{i-1}}, r_i, g_{i,1}, \ldots, g_{i,k-1}) \approx \Pr(g_{i,k} | g_{i-1,n_{i-1}}), \\
(3.8)
\]

i.e. we assume that the root in the current parse depends only on the root of the previous parse, and each inflectional group in the current parse depends only on the last inflectional group of the previous parse (this last assumption is motivated by the remark at the end of Section 2). Now, from (3.6), (3.7), and (3.8) we get:

\[
\Pr(t_i | t_{i-1}) \approx \Pr(r_i | r_{i-1}) \prod_{k=1}^{n_i} \Pr(g_{i,k} | g_{i-1,n_{i-1}}, t_{i-1} | t_{i-1}) \\
(3.9)
\]

where we define \(r_0 = <\text{'.}>'\) and \(g_0, n_0 = <\text{sent}>'.\) Now putting together (3.5) and (3.9) we have:

\[
\Pr(t) \approx \prod_{i=1}^{n} \Pr(t_i | t_{i-1}) \approx \prod_{i=1}^{n} \left[ \Pr(r_i | r_{i-1}) \prod_{k=1}^{n_i} \Pr(g_{i,k} | g_{i-1,n_{i-1}}, t_{i-1} | t_{i-1}) \right]. \\
(3.10)
\]

\(\Pr(r^l | r^m)\) and \(\Pr(g^l | g^m)\) are parameters (root and IG probabilities) which can be estimated using manually disambiguated texts.
3.2 Parameters estimation

Theorem 1. Maximum likelihood estimates for the parameters of the bigram model \( \text{Pr}_b(t) \) are given by:

\[
\text{Pr}_{\text{MLE}}(r^l|r^m) = \frac{C(r^m, r^l)}{C(r^m)}, \quad \text{Pr}_{\text{MLE}}(g^l|g^m) = \frac{C(g^m, g^l)}{C(g^m)},
\]

(3.11)

where \( C(r^m) \) is the number of occurrences of \( r^m \), \( C(r^m, r^l) \) is the number of occurrences of \( r^m \) followed by \( r^l \), \( C(g^m) \) is the number of occurrences of \( g^m \), \( C(g^m, g^l) \) is the number of parses with \( g^m \) as the last IG followed by a parse containing \( g^l \).

Proof. Assume we are observing a sequence of \( n \) tokens \( w_1, w_2, \ldots, w_n \), and each token was manually disambiguated, i.e. we possess a sequence of corresponding parses \( t_1, t_2, \ldots, t_n \). Then the likelihood for our data is given by the equation (3.10), and in order to find maximum likelihood estimates for the parameters \( \text{Pr}(r^l|r^m) \) and \( \text{Pr}(g^l|g^m) \) we need to solve the following optimization problem:

\[
\prod_{i=1}^n \left[ \text{Pr}(r_i|r_{i-1}) \prod_{k=1}^{n_i} \text{Pr}(g_{i,k}|g_{i-1,n_{i-1}}) \right] \quad \rightarrow \quad \text{max}
\]

(3.12)

\[
\sum_l \text{Pr}(r^l|r^m) = 1, \quad \sum_l \text{Pr}(g^l|g^m) = 1.
\]

(3.13)

Using Lagrange multipliers \( \lambda_m \) and \( \beta_m \), seek an unconstrained extremum of the auxiliary function:

\[
\begin{align*}
\ln \left[ \prod_{i=1}^n \left( \text{Pr}(r_i|r_{i-1}) \prod_{k=1}^{n_i} \text{Pr}(g_{i,k}|g_{i-1,n_{i-1}}) \right) \right] \\
&- \sum_m \lambda_m \left( \sum_l \text{Pr}(r^l|r^m) - 1 \right) \\
&\quad - \sum_m \beta_m \left( \sum_l \text{Pr}(g^l|g^m) - 1 \right),
\end{align*}
\]

which is equivalent to:

\[
\begin{align*}
\ln \left[ \prod_{i=1}^n \prod_{k=1}^{n_i} \text{Pr}(r_i|r_{i-1}) \text{Pr}(g_{i,k}|g_{i-1,n_{i-1}}) \right] \\
&- \sum_m \lambda_m \left( \sum_l \text{Pr}(r^l|r^m) - 1 \right) \\
&\quad - \sum_m \beta_m \left( \sum_l \text{Pr}(g^l|g^m) - 1 \right).
\end{align*}
\]
The partial derivative of \( h \) with respect to \( \Pr(r^l|r^m) \) is

\[
\frac{\partial h}{\partial \Pr(r^l|r^m)} = \frac{1}{\Pr(r^l|r^m)} \sum_{i=1}^{n} \delta(r_i, r^l) \delta(r_{i-1}, r^m) - \lambda_m = 0,
\]

where \( \delta \) is the Kronecker delta function, equal to 1 when both of its arguments are same and equal to 0 otherwise. So,

\[
\Pr(r^l|r^m) = \frac{\sum_{i=1}^{n} \delta(r_i, r^l) \delta(r_{i-1}, r^m)}{\lambda_m} = \frac{C(r^m, r^l)}{\lambda_m},
\]

where \( C(r^m, r^l) \) is the number of occurrences of \( r^m \) followed by \( r^l \).

The partial derivative of \( h \) with respect to \( \lambda_m \) is

\[
\frac{\partial h}{\partial \lambda_m} = -\left( \sum_l \Pr(r^l|r^m) - 1 \right) = 0,
\]

and this is equivalent to

\[
\sum_l \Pr(r^l|r^m) = 1 \quad (3.15)
\]

Substitute (3.14) into (3.15) and obtain:

\[
\sum_l \frac{C(r^m, r^l)}{\lambda_m} = 1
\]

Since \( \lambda_m \) does not depend on \( l \), \( \sum_l C(r^m, r^l) = \lambda_m \). Also we have

\[
\sum_l C(r^m, r^l) = C(r^m),
\]

where \( C(r^m) \) is the number of occurrences of \( r^m \).

Thus, \( \lambda_m = C(r^m) \), and therefore we get: \( \Pr_{\text{MLE}}(r^l|r^m) = \frac{C(r^m, r^l)}{C(r^m)}, \) as desired.

Observe that:

\[
\ln \left[ \prod_{i=1}^{n} \prod_{k=1}^{n_i} \Pr(g_{i,k}|g_{i-1,n_{i-1}}) \right] = \sum_{i=1}^{n} \ln \left[ \prod_{k=1}^{n_i} \Pr(g_{i,k}|g_{i-1,n_{i-1}}) \right] = \sum_{i=1}^{n} \sum_{k=1}^{n_i} \ln \left[ \Pr(g_{i,k}|g_{i-1,n_{i-1}}) \right]
\]
The partial derivative of $h$ with respect to $\Pr(g^l|g^m)$ is

$$\frac{\partial h}{\partial \Pr(g^l|g^m)} = \frac{1}{\Pr(g^l|g^m)} \sum_{i=1}^{n} \sum_{k=1}^{n_i} \delta(g_{i,k}, g^l) \delta(g_{i-1,n_{i-1}}, g^m) - \beta_m = 0.$$  

So,

$$\Pr(g^l|g^m) = \frac{\sum_{i=1}^{n} \sum_{k=1}^{n_i} \delta(g_{i,k}, g^l) \delta(g_{i-1,n_{i-1}}, g^m)}{\beta_m} = \frac{C(g^m, g^l)}{\beta_m}, \quad (3.16)$$

where $C(g^m, g^l)$ is the number of parses with $g^m$ as the last IG followed by a parse containing $g^l$.

The partial derivative of $h$ with respect to $\beta_m$ is

$$\frac{\partial h}{\beta_m} = - \left( \sum_l \Pr(g^l|g^m) - 1 \right) = 0,$$

which is equivalent to

$$\sum_l \Pr(g^l|g^m) = 1 \quad (3.17)$$

Substitute (3.16) into (3.17) and obtain:

$$\sum_l \frac{C(g^m, g^l)}{\beta_m} = 1$$

Since $\beta_m$ does not depend on $l$, $\sum_l C(g^m, g^l) = \beta_m$. Also, we have:

$$\sum_l C(g^m, g^l) = C(g^m),$$

where $C(g^m)$ is the number of occurrences of $g^m$.

Thus, $\beta_m = C(r^m)$, and therefore we get: $\Pr_{\text{MLE}}(g^l|g^m) = \frac{C(g^m, g^l)}{C(g^m)}$, as desired. 

However, the maximum likelihood estimates suffer from the following problem: What if a bigram has not been seen in training, but then shows up in the test data? Using the formulas (3.11) we would assign unseen bigrams a probability of 0. Such approach is not very useful in practice. If we want to compare different possible parses for a sentence, and all of them contain unseen bigrams, then each of these parses receives a model estimate of 0, and we have nothing interesting to say about their relative quality. Since we do not want to give any sequence of words zero probability, we need to assign some probability to unseen bigrams.
Methods for adjusting the empirical counts that we observe in the training corpus to the expected counts of n-grams in previously unseen text involve smoothing, interpolation and back-off: they have been discussed by Good [9], Gale and Sampson [8], Written and Bell [21], Knesser and Ney [12], Chen and Goodman [6]. The latter paper presents an extensive empirical comparison of several of widely-used smoothing techniques and introduces a variation of Kneser–Ney smoothing that consistently outperforms all other algorithms evaluated. We believe it should be used for estimating the parameters of the bigram model (3.10).

3.3 Tagging with the Viterbi algorithm

Once parameters are estimated we could evaluate the bigram model (3.10) for all possible parses $t_1^n$ of a sentence of length $n$, but that would make tagging exponential in the length of the input that is to be tagged. An efficient tagging algorithm is the Viterbi algorithm (Algorithm 1). It has three steps: initialization

\begin{algorithm}
\caption{Algorithm for tagging}
\begin{algorithmic}
\Require a sentence $w_{1,n}$ of length $n$
\Ensure a sequence of analyzes $t_{1,n}$
\State $\delta_0(\text{'.'}, <\text{sent}>) = 1.0$
\State $\delta_0(t) = 0.0$ for $t \neq (\text{'.'}, <\text{sent}>)$
\For{$i = 1$ to $n$ step $1$}
\For all candidate parses $t^j$
\State $\delta_i(t^j) = \max_{t^k}[\delta_{i-1}(t^k) \times \Pr_b(t^j|t^k)]$
\State $\psi_i(t^j) = \arg\max_{t^k}[\delta_{i-1}(t^k) \times \Pr_b(t^j|t^k)]$
\EndFor
\EndFor
\State $X_n = \arg\max_{t^j} \delta_n(t^j)$
\For{$j = n - 1$ to $1$ step $-1$}
\State $X_j = \psi_{j+1}(X_{j+1})$
\EndFor
\end{algorithmic}
\end{algorithm}

(lines 1–2), induction (lines 3–8), termination and path readout (lines 9–12). We compute two functions $\delta_i(t^j)$, which gives us the probability of parse $t^j$ for word $w_i$, and $\psi_{i+1}(t^j)$, which gives us the most likely parse at word $w_i$ given that we have the parse $t^j$ at word $w_{i+1}$. A more detailed discussion of the Viterbi algorithm for tagging is provided in [16].
Chapter 4

Conclusion

We reproduced the previous methods of statistical morphological disambiguation [11] for the case of Kazakh language in terms of the Apertium tagset. The data sparseness problem can be reduced by breaking up the morphological analysis across derivational boundaries. The maximum likelihood estimates suffer when we compare possible parses for a sentence that may contain unseen bigrams with zero probability. In order to fix this problem we suggest using Kneser-Ney smoothing technique to estimate the parameters of the bigram model. In order to put our approach into a software it is possible to use one of the open-source language modeling tools (e.g. SRILM1, KenLM2, IRSTLM3) for learning root and IG bigram probabilities, and then implement the provided Viterbi algorithm in any programming language.

Acknowledgment

The author is grateful to the second reader, Aibek Mazakhanov, for careful reading and useful comments.

1http://www.speech.sri.com/projects/srilm/
2https://kheafield.com/code/kenlm/
3https://sourceforge.net/projects/irstlm/
References


