HMM Expanded to Multiple Interleaved Chains as a Model for Word Sense Disambiguation

Denis Turdakov and Dmitry Lizorkin

Institute for System Programming of the Russian Academy of Sciences,
25 Solzhenitsina st., Moscow 109004, Russia
{turdakov, lizorkin}@ispras.ru

Abstract. The paper proposes a method for Word Sense Disambiguation based on an expanded Hidden Markov Model. The method is based on our observation that natural language text typically traces multiple interleaved chains consisting of semantically related terms. The observation confirms that the classical HMM is too restricted for the WSD task. We thus propose the expansion of HMM to support multiple interleaved chains. The paper presents an algorithm for computing the most probable sequence of meanings for terms in text and proposes a technique for estimating parameters of the model with the aid of structure and content of Wikipedia. Experiments indicate that the presented method produces systematically better WSD results than the existing state-of-the-art knowledge-based WSD methods.

1 Introduction

A variety of text analysis tasks essentially rely on word sense disambiguation (WSD) for retrieving semantics from text. Recently, Wikipedia has become an important external knowledge base for performing text analysis tasks, and WSD based on information derived from Wikipedia is currently the area of active research (Grineva *et al.*, 2009; Medelyan *et al.*, 2008; Milne and Witten, 2008; Turdakov and Velikhov, 2008).

The structure of Wikipedia differs (Zlatic *et al.*, 2006) from the structure of the thesaurus WordNet that is traditionally used for WSD. Wikipedia consists of more than 2.5 million articles; each article describes some real world *concept*. Each Wikipedia article has a title that serves as the main *representation* for the concept. The body of an article in particular contains *links* to other conceptually related articles. Formally, each link has two parts: (i) an article in the encyclopedia the link points to and (ii) a *caption* that is displayed to readers of Wikipedia. Besides regular articles, Wikipedia contains several types of special pages. Redirect pages provide *synonyms* of the main representation. Another type of special pages important for this work is a *disambiguation* page, containing a list of articles with a similar ambiguous representation.

Some knowledge-based WSD algorithms (Medelyan *et al.*, 2008; Turdakov and Velikhov, 2008; Milne and Witten, 2008) rely on an assumption that text contains enough unambiguous terms, and these are used as the basis for disambiguation. However, the number of ambiguous representations increases as Wikipedia grows, and additional meanings appear for most commonly used terms. This tendency brings to texts that have a relatively small number of unambiguous terms that weakly correlate with the main subject of text. As a consequence, precision of WSD algorithms based on unambiguous content tends to decrease as Wikipedia continues to evolve.

From this perspective, a WSD algorithm has to be durable to the proportion of ambiguous terms in text. Several research papers thus represent WSD as the maximization problem using the Hidden Markov Model formalism (De Loupy *et al.*, 1998; Molina *et al.*, 2004). Denoting the set of terms in text by T and the set of meanings by M, the input for a WSD algorithm is the sequence of terms $\tau = t_1, \ldots, t_n$ found in text; $t_i \in T$. The maximization problem is finding the most probable sequence of meanings $\mu = m_1, \ldots, m_n$, where $m_i \in M$, in accordance with the model, i.e. $\hat{\mu} = \arg\max_\mu P(\mu \mid \tau) = \arg\max_\mu \left(\frac{P(\mu)P(\tau|\mu)}{P(\tau)}\right)$. Since the probability $P(\tau)$ is constant in the maximization process, the problem is reduced to maximizing the numerator of the equation. For making this equation solvable, the problem is simplified using Markov assumptions. Thus the problem is reduced to solving the equation $\hat{\mu} = \arg\max_\mu \left(\prod_{i=1}^{n} P(m_i \mid m_{i-h:i-1}) \cdot P(t_i \mid m_i)\right)$, where h is the order of the model. The parameters of the last equation define the HMM of order h, where $P(m_i \mid m_{i-h:i-1})$ represents transition probabilities between states, and $P(t_i \mid m_i)$ represents the probability of emitting a term t_i in state m_i, i.e. the probability of emitting a particular term representing the meaning.

Although the WSD task can be easily represented as a maximization problem using the HMM formalism this way, applying HMM to WSD is closely related to the natural language sparseness problem. That is, learning a transition model for WSD requires tremendous amount of training examples—at least several for each pair of concepts. Additionally, the classical Markov assumption itself seems to be not quite suitable for the WSD task; Section 3 gives more detailed considerations on this issue.

To solve the problems outlined above, we propose a model that combines lexical chaining and WSD algorithms and show a possible way to estimating parameters of the presented model.

In summary, the main contributions of this paper are the following:

- We observe that information in natural language text constitutes multiple interleaved chains, and propose an expanded HMM to formalize this observation.
- Based on the proposed model, we present a WSD algorithm and describe a method for estimating parameters of the model using statistical and link information from Wikipedia.

Experiments show that the proposed WSD algorithm produces systematically better results than the state-of-the-art knowledge-based WSD algorithms, thus verifying that the proposed model is appropriate for locating concepts implied in natural language texts.

The rest of the paper is organized as follows. Related work is discussed in Section 2. The proposed approach to word sense disambiguation is motivated by examples in Section 3. The model and the algorithm for the proposed approach are formally described in Sections 4 and 5. Estimation of parameters in order to apply model to WSD is given in Section 6. Results of experimental evaluation are presented in Section 7. We outline future work and conclude in Section 8.

2 Related work

A good survey of WSD algorithms in general is presented in (Agirre and Edmonds, 2007). Due to space limits, here we give a detailed emphasis to recent algorithms that use Wikipedia, HMM or lexical chaining for disambiguation.

Wikipedia has become a popular resource for both knowledge-based and supervised corpus-based WSD methods. Mihalcea and Csomai (2007) use a combination of the Lesk algorithm and Naïve Bayes classifier to produce good disambiguation results. Medelyan *et al.* (2008) and Turdakov and Velikhov (2008) use a combination of semantic relatedness and prior probability of Wikipedia concepts. The best result of Wikipedia-based WSD claimed in literature is presented by Milne and Witten (2008): precision of 98.4% in one of the experiments. Milne and Witten use Wikipedia links to create a training corpus for several machine learning algorithms. However, evaluation is performed over Wikipedia articles themselves as well as training, thus making the evaluation results potentially biased.

Known models for WSD based on HMM use SemCor for training and testing. De Loupy *et al.* (1998) show that SemCor is too small for training even the first-order HMM. De Loupy *et al.* solve the sparseness problem by using WordNet categories to interpolate missing occurrences of meanings. This addition allows increasing precision of their WSD algorithm in comparison to the most common sense. Molina *et al.* (2002, 2004) obtain similar results by using a specialized HMM with states containing additional information about parts of speech of corresponding words.

Mihalcea and Moldovan (2001) propose a WSD algorithm that relies on minimal chaining: it combines a pair of words that have WordNet distance 0 or 1 between some of their senses, and selects these senses. Mihalcea and Moldovan noted that better precision is achieved if both words in a chain are located in text nearby; for an ad-hoc implementation of such a strategy Mihalcea and Moldovan split input documents in their experiment into fragments with at most 15 lines each (Mihalcea and Moldovan, 2001). Our approach provides generalization and formalization of these principles, since in our approach each chain could contain more than two members, and locality within text is modeled through the order of the HMM.

Nelken and Shieber (2007) perform WSD by grouping senses into lexical chains using relations obtained from WordNet. Each chain in their approach is detected independently one by one using a Hidden Markov Model. Although their algorithm is claimed as a generalization of the previously existing ones that rely on lexical chaining, their evaluation reveals the algorithm performing worse than the baseline that simply selects the most common sense for each ambiguous term. Instead of interpreting their evaluation results as a signal for unmotivated assumptions, Nelken and Shieber instead claim this to be a “basic conflict between chaining and WSD”.

Nelken and Shieber (2007) discover that due to sparseness of the two WordNet-based relations they are using, lexical chaining applied to WSD is unlikely to outperform the most common sense baseline. From this analysis Nelken and Shieber conclude a too general statement: “not to use chaining algorithms for WSD”. In this paper we show the incorrect generality of these statements, and the most common sense baseline is considerably outperformed by the method we propose.

3 Motivating Examples

Recent research papers show that it is practically complicated to learn the transition model even for a first-order HMM if applied to the WSD problem. But even supposing it is possible to compute parameters for HMM of any order, is the classical HMM still an adequate model for representing a sequence of meanings for terms encountered in natural language text? Let us have a look at some practical examples.

Example 1. Imagine a news article about sports medicine: a novel way of treating professional diseases with the aid of a drug never used for this purpose before. Potential terms encountered by a WSD system within such text could be: *football*, *The drug*, *sports medicine*, etc. Since the particular drug is reported to be used in the sports field for the first time, it is likely that the name of this drug (called ‘*The drug*’ here for illustration) is not related to sports concepts in a knowledge base used for training the system, i.e. $P(\textit{The drug} \mid \textit{football}) = 0$. In the case of classical HMM, the system would need to rely on some heuristics or smoothing methods to find the correct sequence of meanings. On the other hand, if a model supports several chains, then two chains are formed, and the probability of this event is equal to the joint probability of two independent meanings. Moreover, these chains could be merged into one when processing further terms in input text, e.g. when encountering the term *sports medicine* that covers both topics.

Example 2. Let us have a look at the example we came across while investigating the applicability of a classical HMM to WSD. This example presents a fragment of a real life news article about professional footballers and their cars. In the sentence

Cristiano Ronaldo hit the *headlines* when he crashed his *Ferrari*

three terms from a Wikipedia dictionary are emphasized: *Cristiano Ronaldo*, *headline* and *Ferrari*. The term *Ferrari* is an ambiguous one and has at least two meanings: (a) Italian sports car manufacturer and (b) Matteo Ferrari, an Italian football player. Given the strong football-related context represented by *Cristiano Ronaldo*, most WSD algorithms including classical HMM-based ones tend to choose the footballer rather than the car brand as the meaning for *Ferrari*, even though the car brand is a more commonly used meaning. On the other hand, a model that supports multiple chains would find it more probable to start a new chain consisting of the most common meaning in this case, and the other car-related meanings are added to the chain if appropriate terms appear in text of the news article further.

It can be observed from the above examples that natural language texts typically describe multiple topics or multiple aspects within a topic, and these are not properly reflected by a classical HMM. Our goal is thus to design a model that would provide a natural way for representing meaning of text as a set of interleaved Markov chains. With this goal in mind, we suggest expanding the HMM formalism to the case of multiple interleaving chains. First, the expanded model is described in general, and then the application of the model to the WSD task is discussed.

4 Expanding HMM

Like in the classical HMM, we discover stationary processes modeled as Markov chains of some order m. The main distinction between the classical model and the one presented in this section is that a current state is now allowed to either become a part of some existing chains or to start a new chain.

In the following, we denote states by regular mathematical letters, the state of the system at the k-th step denoted as S_k. Markov chains are denoted by calligraphic letters, e.g., $\mathcal{L}$, $\mathcal{N}$. We use overlining to combine components that constitute a single chain, thus $\overline{\mathcal{L}S_k}$ denotes a chain consisting of consequent states of $\mathcal{L}$ and the last state S_k. Additionally, notation $S_k \in \mathcal{L}$ means that the state S_k belongs to the chain $\mathcal{L}$. Finally, the hat notation $\widehat{S_iS_j}$ is used to combine a pair of states S_i, S_j that belong to a common chain.

The proposed expanded model is formalized in two phases. First, we consider the case when all previous states of the system constitute a single chain. Then, this particular case is extended to the general case of multiple chains.

4.1 Previous states constitute a single chain

With all the previous states $S_1, S_2, \ldots, S_{k-1}$ combined into a single chain $\mathcal{L}$, this case is nevertheless different from the classical HMM in the way the current state is handled. Namely, two options for the current state S_k are possible: (a) either S_k joins $\mathcal{L}$ as a new state, or (b) S_k does not belong to $\mathcal{L}$ and forms a new chain $\mathcal{N}$. Probabilities of these options are written correspondingly as:

$$P(\overline{\mathcal{L}S_k}) = P(\mathcal{L}) \cdot P(S_k \in \mathcal{L}) \cdot P(S_k \mid \mathcal{L}) \ , \quad (1)$$

$$P(\mathcal{L}, \mathcal{N}) = P(\mathcal{L}) \cdot P(S_k \notin \mathcal{L}) \cdot P(S_k) \ . \quad (2)$$

The right-hand side of each equation consists of exactly three multipliers; we will further refer to these multipliers as the *first*, the *second* and the *third* ones respectively. Each of the equations (1) and (2) expresses the total probability of each of the options; thus both equations include probability of the chain $\mathcal{L}$ as their first multipliers. The second multiplier specifies the probability of whether the current state S_k belongs to the chain. If S_k joins the chain, then the third multiplier in equation (1) expresses the probability of S_k appeared given the chain. If S_k forms a new chain, then the third multiplier in equation (2) does not depend on the chain $\mathcal{L}$.

Our goal is to locate the most probable sequence of states and their grouping into chains; this implies computing the probabilities $P(\overline{\mathcal{L}S_k})$ and $P(\mathcal{L}, \mathcal{N})$. Further investigating equations (1) and (2), our next step is to work out the formulae for computing each of the three multipliers.

The *first* multipliers in (1) and (2) are computed recursively over the length of a chain. For discovering the *second* multipliers, we make an assumption similar to the Markov assumption:

Assumption 1 *Probability of a current state S_k belonging to a chain $\mathcal{L}$ depends only on a fixed finite number of previous states $S_{k-1}, S_{k-2}, \ldots, S_{k-h}$ in $\mathcal{L}$.*

We further refer to states $S_{k-1}, \ldots, S_{k-h}$ from Assumption 1 as *active* ones; we refer to a chain that contains active states as an *active chain*. Note that the number of active chains is less than or equal to the number of active states. In this paper we consider only finite *history* of active states, so in addition to the order m of the classical Markov Model for linked chains, we introduce the *order of the expanded model*, further denoted as h.

With Assumption 1, the second multipliers in equations (1) and (2) are expressed through a relationship between S_k and the set of active states in $\mathcal{L}$. To further simplify the computation of the second multipliers, we express them through a pair-wise relationships between states, by introducing one more assumption:

Assumption 2 *For different states S_i, S_j and S_k, the event "S_i and S_k belong to a common chain" is independent of the event "S_j and S_k belong to a common chain". That is, for $i \neq j$, $i \neq k$ and $j \neq k$:* $\boldsymbol{P(\widehat{S_iS_k}}$ ***and*** $\boldsymbol{\widehat{S_jS_k}) = P(\widehat{S_iS_k}) \cdot P(\widehat{S_jS_k})}$ **.**

Applying Assumptions 1 and 2 and denoting the set of active states in the chain $\mathcal{L}$ as $\Omega = \{S_{k-1}, \ldots, S_{k-h}\}$, the second multipliers in equations (1) and (2) are computed as probabilities of complimentary events: $P(S_k \notin \mathcal{L}) = \prod_{S_i \in \Omega}[1 - P(\widehat{S_iS_k})]$, $P(S_k \in \mathcal{L}) = 1 - \prod_{S_i \in \Omega}[1 - P(\widehat{S_iS_k})]$. The *third* multipliers in equations (1) and (2) can be learnt from a tagged corpus as their analogues from a classical Markov model. A particular way of defining transition probability $P(S_k \mid S_{k-1}, \ldots, S_{k-m})$ as well as probability $P(\widehat{S_iS_k})$ depends on a particular domain the model is applied to. Section 6 illustrates a way of computing these probabilities for an application of the proposed model to the WSD task based on Wikipedia.

4.2 Multiple chains

In the general case of multiple chains, let us denote the set of all chains as $\Lambda = \{\mathcal{L}_1, \ldots, \mathcal{L}_q\}$. The current state S_k can belong to zero or more chains in Λ. The current state S_k belonging to more than one chain means that S_k merges these chains into one. Formally, for an arbitrary subset of chains $\lambda \subset \Lambda$, $\lambda = \{\mathcal{L}_{i_1}, \mathcal{L}_{i_2}, \ldots, \mathcal{L}_{i_r}\}$, the probability of S_k belonging to exactly this subset is written by combining equations (1) and (2):

$$P(\overline{\lambda S_k}, \Lambda \setminus \lambda) = P(\Lambda) \cdot P(S_k \in \lambda, S_k \notin \Lambda \setminus \lambda) \cdot P(S_k \mid \lambda) \ . \tag{3}$$

Here, notation $S_k \in \lambda$ stands for the event of S_k belonging to every chain in λ; notation $S_k \notin \Lambda \setminus \lambda$ stands for the event of S_k belonging to neither chain in $\Lambda \setminus \lambda$.

Like in case of a single chain, the first multiplier in (3) is computed recursively over k. The second multiplier in (3) is computed using independence of individual chains:
$P(S_k \in \lambda, S_k \notin \Lambda \setminus \lambda) = \prod_{L_i \in \lambda} P(S_k \in \mathcal{L}_i) \times \prod_{L_j \in (\Lambda/\lambda)} P(S_k \notin \mathcal{L}_j)$.

If λ consists of exactly one chain, the third multiplier in (3) is computed in the same way as discussed in the previous subsection. Since the general case of λ containing multiple chains has the effect of all these chains being merged together, probability of S_k given λ is defined as probability of S_k given the merged chain: $P(S_k \mid \mathcal{L}_{i_1}, \ldots, \mathcal{L}_{i_r}) = P(S_k \mid \mathcal{L})$, $\mathcal{L} = \bigcup_{j=1}^{r} \mathcal{L}_{i_j}$. For instance, if order of Markov model for chains is $m = 2$, and state S_k belongs to both chains $\mathcal{L}$ and $\mathcal{N}$ (Figure 1a), then $P(S_k|\mathcal{L}\mathcal{N}) = P(S_k|S_{k-1}S_{k-2})$. If S_k belongs only to $\mathcal{L}$, then $P(S_k|\mathcal{L}\mathcal{N}) = P(S_k|\mathcal{L}) = P(S_k|S_{k-2}S_{k-4})$, and so on. Note that we do not need to recompute chains, we just need to compute probability of a new state given the last m states that belong to chains in λ.

For the classical Markov Model, the most likely sequence of states given a sequence of observation can be found using the Viterbi algorithm. The Viterbi algorithm bases on a recursive relationship between the most likely path to each state S_{k+1} and the most likely path to each previous state S_k. In the next section, we define the similar algorithm for the expanded model.

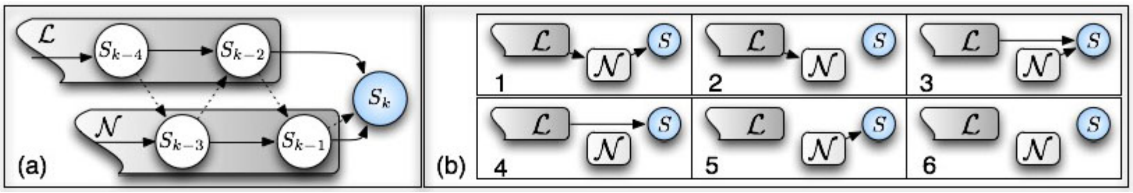

Figure 1: Possible connections between chains and states: (a) New state S_k belongs to chains $\mathcal{L}$ and $\mathcal{N}$; (b) Connections between chains $\mathcal{L}$ and $\mathcal{N}$ and a new state S.

5 Algorithm for finding the most probable sequence of states

Algorithm 1 for detecting the most likely sequence of states for a given sequence of observations is similar to its analogue for the classical Markov model except for the function $computePath$. The function $computePath$ accepts paths to previous states and a new state corresponding to the current observation and computes the most probable path to this new state; the definition of the function is provided below. Paths through previous states are stored in the associative container $prevPath$ (lines 2, 7, 9 of the Algorithm). The function $combination(s^j_{i-h\ldots i-1})$ called in line 5 of the Algorithm produces a set of all combinations of states corresponding to observations $i-h,\ldots,i-1$. Note that in the case of equiprobable paths the algorithm prefers a path consisting of the smallest number of chains.

Algorithm 1 Compute most likely explanation

```
Input: $h$, sequenceOfObservation
Output: mostProbableSequenceOfStates
 1: for $s^j_1 \in$ {states corresponding to first observation} do
 2:     prevPath[$s^j_1$]=$s^j_1$
 3: for $i$ = 2 to $n$ do
 4:     for $s^j_i \in$ {states corresponding to $i$th observation} do
 5:         for all $u \in combination(s^j_{i-h\ldots i-1})$ do
 6:             if $i <= k$ then
 7:                 prevPath[$u \cup s^j_i$] = computePath(prevPath[$u$],$s^j_i$)
 8:             else
 9:                 prevPath[$u/\{s_{i-h}\} \cup s^j_i$] = $\arg\max_{s_{i-h}}$ ($computePath$(prevPath[$u$], $s^j_i$)
10: return $\arg\max_{u=\{u_{n-h\ldots n}\}}$ prevPath($u$)
```

The function $computePath$ is defined separately for two different cases. Namely, since the probability of a new state S_k joining different chains potentially changes with respect to λ in both the second and the third multipliers in (3), it is instructive to investigate the effect each of the multipliers has individually. For achieving this goal, special consideration is given to a zero-order HMM for chains (i.e. to the case of $m = 0$), since in this case $P(S_k \mid \lambda) = P(S_k)$ and thus the third multiplier in (3) is constant with respect to λ. We refer to this particular case of the expanded model as a *weak* model and discuss it first. For the general case of $m > 0$, we refer to the model as a *full* one and discuss it in the subsequent subsection.

5.1 Weak model

With the weak model having $m = 0$, the right-hand side of equation (3) differs only in its second multiplier for different λ. Such uniformity of equation (3) for the weak model makes it possible to save computational effort in locating the most probable sequence of states.

We present two propositions to reduce the search space for the most probable path; then the function $computePath$ is defined. For illustrative purposes, let us consider the case of Λ consisting of only two chains: $\Lambda = \{\mathcal{L}, \mathcal{N}\}$. Figure 1b lists all possible connections between $\mathcal{L}$

and $\mathcal{N}$ and the current state S. The first two cases in the figure represent the chain $\mathcal{N}$ being the continuation of the chain $\mathcal{L}$; the remaining cases in the figure represent $\mathcal{N}$ being a separate chain.

Let the chain $\mathcal{N}$ consist of states $S_{i_1}, \ldots, S_{i_p}$. The first proposition affirms that the current state does not break previously established connections in the most probable path.

Proposition 1 *If a chain $\mathcal{N}$ is a part of the most probable path, and the first state S_{i_1} of the chain $\mathcal{N}$ belongs to $\mathcal{L}$ with probability greater than $\frac{1}{2}$, then $\mathcal{L}$ is a part of the most probable path as well, and $\mathcal{N}$ is the continuation of $\mathcal{L}$.*

In terms of Figure 1b, Proposition 1 states that (i) it follows from $P(Case_1) \geq P(Case_2)$ that $P(Case_1) \geq P(Case_i)$, $i = \overline{1..6}$; and (ii) it symmetrically follows from $P(Case_2) \geq P(Case_1)$ that $P(Case_2) \geq P(Case_i)$, $i = \overline{1..6}$. Due to space limits we omit proofs in this paper.

The second proposition establishes that the current state in the most probable path can affect connections among existing chains, but only among active ones.

Proposition 2 *If $\mathcal{L}$ and $\mathcal{N}$ are disjoint active chains (cases 3-6 in Figure 1b), there can exist state S such that the most probable path to S contains $\mathcal{L}$ and $\mathcal{N}$ as a merged chain (case 1 in Figure 1b).*

Generalization of both propositions to the case of multiple chains is made by induction. Proposition 1 verifies that there is no need to consider once merged chains as disjoint ones. Proposition 2 shows that disjoint chains can still get merged, but merging can only take place while chains remain active. The combination of these propositions allows reducing the search space while finding the most probable path. The respective definition of the function $computePath$ is given in Algorithm 2.

Algorithm 2 Function $computePath$ for weak model

```
Input: previousPath, newState
Output: newPath
$chainsSet = getChainsCombinations(previousPath)$
for all $joinedChain \in chainsSet$ do
    Compute $P(newState \in joinedChain)$
return $\arg\max_{processed\ paths} P(path)$
```

The local variable $chainsSet$ in Algorithm 2 is a set containing all possible combinations of active chains. With n denoting the number of active chains, the number of items in the set $chainSet$ is calculated as $|chainsSet| = \sum_{k=0}^{n} S(n,k) = B_n$, where $S(n,k)$ is a Stirling number and B_n is a Bell number.

5.2 Full model

For the full model, the third multiplier in equation (3) essentially depends on connectivity of active chains, and thus Propositions 1 and 2 do not hold for the full model. For instance, the probability $P(S \mid \mathcal{L}\mathcal{N})$ in Case 1 in Figure 1b can be either greater than or less than the probability $P(S \mid \mathcal{L})$ in Case 4.

For each path, the algorithm thus has to process all possible active chains that could potentially be constructed. The definition for the function $computePath$ is the same as the one given in Algorithm 2, with the difference that $chainsSet$ is now the set of *all* possible combinations of active states into chains. The number of such combinations is given by the Bell number B_n as well, but with n now standing for the number of active *states* in the path.

Comparative evaluation of weak and full models is presented in Section 7.

6 Application to WSD

For applying the proposed model to WSD, we construct a dictionary that contains representations for all concepts from the English Wikipedia. Performing WSD over input text implies: (a) finding all phrases in text that match representations from the constructed dictionary, and (b) selecting the appropriate Wikipedia concept for each matched representation.

As a preparatory offline phase, we collect titles of all regular Wikipedia articles and of all redirect pages. Additional associations between concepts and representations are retrieved from Wikipedia disambiguation pages and from link captions.

To specialize the proposed model to WSD, we now estimate the three parameters of the model: 1) the probability $P(\widehat{m_i m_j})$ of two states belonging to the same chain; 2) the transition model $P(m_i \mid m_{i-h:i-1})$; and 3) the observation model $P(t_i \mid m_i)$.

With respect to the Wikipedia dictionary used for WSD in this paper, we suggest to base the estimates on semantic relatedness between Wikipedia concepts. Precisely, we chose the neighborhood-based measure presented in (Turdakov and Velikhov, 2008).

1. For estimating the *probability of two states belonging to the same chain*, the following heuristics is introduced:

Heuristics 1 *Probability of two states belonging to the same chain is a function of semantic relatedness:* $P(\widehat{m_1 m_2}) = \phi(rel(m_1, m_2))$.

We learnt the function ϕ fully automatically from a collection of un-annotated documents. Namely, due to certain correlation discovered in related work between lexical chaining and clustering within a graph of disambiguated concepts (Medelyan, 2007), we utilized this observation for learning ϕ. Each document in the collection was represented as a weighted graph, with vertices for concepts and edges between vertices having weight equal to semantic relatedness between these concepts. Note that we kept all alternative concepts for each term located in a document. Next, a clustering algorithm (Clauset *et al.*, 2004) was applied to each graph, and two concepts were considered to belong to the same lexical chain if and only if the corresponding vertices in the graph fell into the same cluster. The set of positive examples for training thus consisted of concept pairs from a common cluster; negative examples consisted of concept pairs that belonged to a common document, but to different clusters. Obtaining the set of 137,324 positive and 859,076 negative examples, the function ϕ was computed in the space of step functions with step 0.01.

2. The *transition model* is estimated using the following heuristics:

Heuristics 2 *The probability of a meaning* m_i *given the sequence of previous meanings* m_{i-1}, m_{i-2}, ..., m_{i-h} *is proportional to a linear combination of (i) semantic relatedness between the current and the previous meanings and (ii) the prior probability of the current meaning:*
$P(m_i \mid m_{i-1}) = \alpha \cdot (rel(m_i \mid m_{i-h:i-1}) + \beta \cdot P(m_i))$.

Since the coefficient α does not affect the result of the maximization problem (recall Algorithm 1), we can remove α from further consideration. Also, in case of the current meaning m_i not depending on previous meanings and being the first member of a new chain, the right-hand side of the equation should reduce to prior probability for consistency, thus giving $\beta = 1$.

Using semantic relatedness to estimate parameters of the model helps to solve the sparseness problem, because relatedness can be calculated for any pair of Wikipedia concepts.

3. The *observation model* and the prior probability of meanings are estimated from the Wikipedia corpus statistically using concept names, names of redirects and link captions.

Applying the model to a sequence of representations located in input text yields the most probable sequence of corresponding meanings, i.e. WSD for input text. Quality of results produced by the suggested WSD algorithm is evaluated in the next section.

7 Evaluation

WSD algorithms based on Wikipedia traditionally use Wikipedia articles themselves and their link information for evaluation. Namely, for each term appearing as a link caption and presented in a dictionary, its corresponding meaning is the one pointed by the link. It can easily be observed that evaluation on Wikipedia articles is potentially biased towards algorithms that use machine learning to train on the Wikipedia corpus itself (Milne and Witten, 2008).

We used a collection of Wikipedia articles composed of 500 random regular articles in order to compare our method with the existing WSD algorithms. In addition, we took the test set presented in (Milne and Witten, 2008), annotated with the aid of the Mechanical Turk service. Finally, we manually annotated a collection of 131 documents consisting of news articles from several online sources and of several scientific papers. Statistics for these collections is presented in Table 1. Note that the proportion of ambiguous terms in the manually annotated collection is higher than in the collection of Wikipedia articles.

Table 1: Statistics for test collections

	Wikipedia articles	Milne and Witten (2008)	News & scientific papers
#documents	500	50	131
#terms	50947	727	8236
#ambiguous terms	39332	479	6952
#meanings (avg)	35.34	29.94	22.34

We use the Wikipedia dump of March 2009 for the evaluation. Results are presented in Table 2. We apply WSD algorithms to terms that have proper concepts in Wikipedia, so precision and recall are equal. Also, we skip terms that have no proper concepts in Wikipedia while annotating the test collection to exclude the influence of insufficient coverage of Wikipedia on the results.

Table 2: Precision (%) for different WSD algorithms for the test collections

	Wikipedia articles	Milne and Witten (2008)	News & scient. papers
Turdakov and Velikhov (2008)	85.12	78.81	64.34
Most Common Sense	90.10	85.33	67.61
HMM-1	90.13	85.33	67.61
HMM-2	91.51	85.33	67.72
$h = 2$, $m = 0$	93.36	89.00	74.75
$h = 2$, $m = 1$	93.68	89.18	75.10
$h = 2$, $m = 2$	92.87	88.80	73.93
$h = 3$, $m = 0$	93.67	88.41	75.60
$h = 3$, $m = 1$	93.72	**89.38**	75.56
$h = 3$, $m = 2$	**93.78**	**89.38**	**76.13**

The results shown by our model are significantly better than the ones produced by the classical HMM, even in case of the zero order HMM for chains. Evaluation also shows that the weak model produces good results, just little bit worse than the full model demonstrates.

Precision of 76.4% reported by Milne and Witten (2008) on their testing corpus is lower than for our algorithm; however, their figure also incorporates precision of link detection, thus preventing direct comparison of the two approaches. Although the algorithm by Milne and Witten relies on extensive machine learning, we believe that our algorithm provides quite comparable precision.

8 Conclusion and future work

This paper introduces HMM expansion to several interleaved chains and applies the proposed model to word sense disambiguation. The accompanying result achieved by the proposed algorithm is segmentation of disambiguated concepts from input text into lexical chains. The proposed algorithm produces considerably better results than the classical HMM applied to WSD.

A possible way to improve results produced by the algorithm is to investigate more sophisticated strategies for choosing active states. Assumption 1 made in the paper is weaker than the Markov assumption, for our approach makes it possible to keep some of the states active for using them in any part of input text. For example, it could be reasonable to experiment with keeping concepts from the text title active. Such a modification would make topics introduced in the title active for the whole text to construct chains that describe main topics.

Another way to improve WSD results is to discover the function ϕ that establishes the probability of two terms belong to one chain with respect to their semantic relatedness. This function is the primary parameter of the algorithm that affects the length of a chain. In this paper, we suggested one way for computing this function; however, some other way may fit WSD better, and such analysis could be a good subject for future work.

References

Agirre, E. and P. Edmonds, editors. 2007. *Word Sense Disambiguation: Algorithms and Applications*. Text, Speech and Language Technology. Springer.

Clauset, A., M. E. J. Newman, and C. Moore. 2004. Finding community structure in very large networks. *Physical Review E*, 70:066111.

De Loupy, C., M. El-Beze, and P. Marteau. 1998. Word sense disambiguation using HMM tagger. In *1st Int. Conf. on Language Resources and Evaluation, LREC*, pages 1255–1258.

Grineva, M., M. Grinev, and D. Lizorkin. 2009. Extracting key terms from noisy and multitheme documents. In *18th Int. Conf. on World Wide Web*, pages 661–670. ACM.

Medelyan, O. 2007. Computing lexical chains with graph clustering. In *ACL*.

Medelyan, O., I. H. Witten, and D. Milne. 2008. Topic indexing with Wikipedia. In *1st AAAI Workshop on Wikipedia and Artificial Intelligence*.

Mihalcea, R. and A. Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *16th ACM Conference on Information and Knowledge Management*, pages 233–242. ACM.

Mihalcea, R. and D. Moldovan. 2001. A highly accurate bootstrapping algorithm for word sense disambiguation. *International Journal on Artificial Intelligence Tools*, 10(1-2):5–21.

Milne, D. and I. Witten. 2008. Learning to link with Wikipedia. In *17th ACM Conference on Information and Knowledge Management*, pages 509–518. ACM.

Molina, A., F. Pla, and E. Segarra. 2004. WSD system based on Specialized Hidden Markov Model. In *Senseval-3*, pages 171–174. Association for Computational Linguistics.

Molina, A., F. Pla, E. Segarra, and L. Moreno. 2002. Word sense disambiguation using statistical models and WordNet. In *Language Resources and Evaluation (LREC)*.

Nelken, R. and S. Shieber. 2007. *Lexical chaining and word-sense-disambiguation.* Technical Report TR-06-07, School of Engineering and Applied Sciences, Harvard University.

Turdakov, D. and P. Velikhov. 2008. Semantic relatedness metric for Wikipedia concepts based on link analysis and its application to word sense disambiguation. In *SYRCoDIS*.

Zlatic, V., M. Bozicevic, H. Stefancic, and M. Domazet. 2006. Wikipedias: Collaborative web-based encyclopedias as complex networks. *Physical Review E*, 74:016115.