ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **19**, Numbers 1-2, 2016, 78–97

An Overview of Data-Driven Part-of-Speech Tagging

Dan TUFIŞ

Research Institute for Artificial Intelligence "Mihai Drăgănescu",
Romanian Academy
E-mail: tufis@racai.ro

Abstract. Over the last twenty years or so, the approaches to part-of-speech tagging based on machine learning techniques have been developed or ported to provide high-accuracy morpho-lexical annotation for an increasing number of languages. Given the large number of morpho-lexical descriptors for a morphologically complex language, one has to consider ways to avoid the data sparseness threat in standard statistical tagging, yet to ensure that the full lexicon information is available for each word-form in the output. The paper overviews some of the major approaches to part-of-speech tagging and touches upon the tagset design, which is of crucial importance for the accuracy of the process.

Key-words: ambiguity class, data sparseness, lexical ambiguity, machine learning, multilinguality, part-of-speech tagging, tagset design.

1. Introduction

Lexical ambiguity resolution is a procedure by which a computer program reads an arbitrary text, segments it into tokens and attaches to each token the information characterizing the lexical and contextual properties of the respective word. This information can be explicitly specified or encoded in a compact way by a uniquely interpretable label. Such a description is called a part-of-speech tag (or POS tag for short).The set of all possible tags is called the tagset of the lexical ambiguity resolution process.

We use here the notion of token to refer to a string of characters from a text that a word-identification program would return as a single processing unit. The procedure that figures out what the tokens are in the input is called tokenizer

or lexical segmenter. The complexity of this process varies depending on the language family, and tokenization for many languages is a research topic in itself (e.g. for Asian languages which do not use white spaces between words, agglutinative languages or even for compound productive languages). We do not address in this paper the problem of tokenization, but the reader should be aware that tokenizing a sentence might not merely be white-space identification and a multi-word dictionary lookup. Lexical ambiguity resolution is applied to the tokenized input, in order to assign the appropriate POS tag to each token. The POS assignment is achieved by a program called POS tagger. An ambiguous lexical item is one that may be classified differently depending on its context. The set of all possible tags/descriptions a lexical token may receive is called the (lexical) ambiguity class (AC) of that token. In general, many tokens may share the same ambiguity class. Information about the ambiguity class of each lexical token, the probabilities of the tags in an ambiguity class as well as the interdependencies between tokens' tags are knowledge sources for the tagger's decision making. The majority of the available POS taggers have this a-priori knowledge constructed (at least partially) in a statistical manner. Let us call a language model (LM) all the a-priori information a tagger needs for its job. The construction of a language model may be achieved manually by human experts who write dictionary entries, and grammar rules to decide the right interpretation in context for ambiguous lexical items. This approach is sometimes called introspection driven. Another option is the data-driven one, where a specialized part of the tagger (called the learner) learns the language model from the training data. In this article we will discuss only some of the data-driven models and technologies used in tagging.

Depending on the way a data-driven language model is created/learned, one distinguishes two major approaches: supervised versus unsupervised methods. As usually happens with dichotomy classifications, there are also mixture approaches, called semi-supervised or partially-supervised methods. The supervised learners draw the LMs from annotated corpora (sometimes supplemented by human-made lexicons), while the unsupervised learners rely only on raw corpora with or without a lexicon that specifies the ambiguity classes for the lexical stock (some approaches use only a few frequent lexical entries as learning seeds). Although for most of the national languages annotated corpora do exist, this is definitely not the case for many other languages or dialects of interest for researchers and even commercial developers. Because annotated corpora are expensive to build, the interest in unsupervised learning for POS tagging significantly increased lately. For an interesting account on unsupervised learning research for POS tagging see [11].

Among the data-driven models for POS tagging, in this paper we will discuss only the N-gram models with the Hidden Markov Models (HMM) as the most used representatives and Maximum Entropy Models (ME) with the more recent Conditional Random Fields Models (CRF). There are several other types of tagging models [28], most of them data-driven, but due to space limitation they are not dealt with here. However, for the interested reader we provide some

reference points: Decision-trees [5], [48]; Neural Networks [15], [6]; Bayesian Nets [38]; Case-Based [19], Inductive Logic Programming [17]; Support Vector Machine [29] etc.

The paper is organized as follows: first, we will discuss the basic model, the N-grams (Section 2). Then, we will address a general problem for any data-driven model, namely the data-sparseness and various ways to mitigate its consequences (Section 3). Section 4 introduces the generative/discriminative dichotomy for the statistical tagging models and discusses the HMM generative models and the ME and CRF discriminative models. Section 5 discusses an effective method to use large lexical tagsets designed for interoperability in a multilingual environment.

2. N-gram Models

The tagging problem is defined as the assignment to each word in an input sequence w_1, w_2,...,w_n a unique label representing the appropriate part-of-speech interpretation, thus getting the output w_1:t_1, w_2:t_2,...,w_n:t_n. As the sequence of words and their associated tags is arbitrary, it is convenient to describe the process in statistical terms: let us consider a text of k words as a sequence of random variables X_1, X_2, ..., X_k, each of these variables taking arbitrary values from a set V called the lexicon. In a similar way, we consider the sequence of tags attached to the k words of a text as a sequence of k variables Y_1, Y_2, ..., Y_k, each of them taking values from the tagset T. Our linguistic intuition says that once a variable X_i has been instantiated by a specific word w_i, the value of the associated tag t_i is restricted to one of its ambiguity class. That is $t_i \in AC(w_i)=\{t_{i1}, t_{i2}, ..., t_{im}\}$. However, not all of them are equally likely. We might write this by using different probabilities of the assignment: $p(w_i|t_{i1})$, $p(w_i|t_{i2})$ ... $p(w_i|t_{im})$. Such probabilities can be easily estimated from an annotated corpus by counting how many times the tag t_{ik}has been assigned to the word w_i out of the total number of occurrences of the tag t_{ik}:

$$\tilde{p}(w_i|t_{ik}) = \frac{count(w_i : t_{ik})}{count(t_{ik})} \tag{1}$$

Going one step further, the linguistic intuition says that deciding on the appropriate tag from the ambiguity class of a given word, depends on its context. Let us call the context of the tag t_i, $C(t_i) =<t_{i-1}, t_{i-2}, ..., t_1 >$, the sequence of tags assigned to the words preceding the word w_i. For the development of tagging technology, two major assumptions are instrumental: *the tag of a given word depends on the word itself and on the tags of its context.* Assuming that the context dependency of a tag is restricted to a number of N-1 preceding tags, one can make the following estimations:

$$\tilde{p}(t_i|t_{i-1}\, t_{i-2} \ldots t_{i-N+1}) = \frac{count(t_i t_{i-1} t_{i-2} \ldots t_{i-N+1})}{count(t_{i-1} \ldots t_{i-N+1})}. \tag{2}$$

However, this approach is impractical because of the lack of tagged data to cover the huge number of possible contexts for all tags (T^N). Most N-gram models, for *tractability* reasons, limit the contexts by considering only one or two preceding words/tags:

$$\tilde{p}(t_i|t_{i-1}) = \frac{count(t_i t_{i-1})}{count(t_{i-1})} \quad \text{or} \quad \tilde{p}(t_i|t_{i-1}t_{i-2}) = \frac{count(t_i t_{i-1} t_{i-2})}{count(t_{i-1}t_{i-2})}. \qquad (3)$$

Assuming that the probabilities $\tilde{p}(w_i|t_i)$ and $\tilde{p}(t_i|t_{i-1})$ are reliably estimated, one may solve the problem of assigning an input string w_1, w_2, ..., w_n the part-of-speech annotation w_1:t_1, w_2:t_2, ... , w_n:t_n. What is desired is that this annotation should be the most linguistically appropriate, *i.e.* the tags assignment should have the highest probability among a huge number of possible annotations:

$$w_1 : t_1, w_2 : t_2, ..., w_N : t_N = argmax_{t_1 ... t_N} \prod_{i=1}^{N} \tilde{p}(w_i|t_i)\tilde{p}(t_i|t_{i-1}). \qquad (4)$$

Finding the most likely tags assignment for the input string is an optimization problem and it can be done in different ways, such as: using a sliding window of k<N words and choosing the best assignment in the window [54] using (binary) decision trees [48] or by dynamic programming [12]. The sliding window approach is very simple and very fast, but it might not find the global (sentence level) optimal solution. The decision tree–based search is more flexible (allows for variable length of contexts, including negative restrictions) but might also get stuck into a local maximum. Dynamic programming, with its most acknowledged Viterbi algorithm [58], relies on fixed length contexts but it is guaranteed to find the global optimal solution (*assuming* that the conditional independences hold and *assuming* that all conditional probabilities are reliably estimated).

3. Data sparseness

This is an unavoidable problem for any model that makes probability estimations based on corpus counting. Equation (4) shows a product of probability estimates which would be zero in case the input string contains one or more words or tag sequences unseen in the training corpus. This would render such an input string as impossible (p=0) which certainly is wrong. On the other hand, to be reliable, the estimations should be made based on a minimal number of observations, empirically say 5 or 10. Yet, this is practically impossible, because whatever large training corpus one reasonably assumes, according to *Zipf Rank-Frequency Law* there will exist always a long tail of rare words. Avoiding the "*data sparseness curse*" in POS tagging requires finding a way to associate

whatever unknown word from an input string with the most likely tags and their probability distribution and to use the knowledge acquired from the training corpus to estimate probabilities for tag bigrams/trigrams not seen during the training.

A naïve method to treat the *out of the dictionary words* (ODW-words not seen during the training phase) is to automatically associate as an ambiguity class the list of tags for open class categories (nouns, verbs, adjectives, adverbs) and their probabilities uniformly distributed. A better approach is to do a morphological analysis of the ODWs or a simpler "prefix"/"suffix" investigation. Here "*prefix*" and "*suffix*" is used in a loose way, namely as a string of k letters that start or end an input word. For instance, capitalization of a sentence internal word might be a strong clue that it is a noun (in German) or a proper noun (in most of the languages). In languages with significant inflectional morphology the suffixes carry a lot of information that might help in inducing a reasonably reduced ambiguity class. Tag probabilities are assigned according to the frequency of the endings observed in the training corpus. For instance, words in Wall Street Journal part of the Penn Treebank ending in *able* are adjectives in 98% of the cases (fashionable, variable), with the rest of 2% being nouns (cable, variable) (Brants 2000). The probability distribution for a particular suffix may be generated from the words in the training corpus that share the same suffix. A usual assumption is that the distribution of unseen words is similar to the distribution of rare words and consequently considering from the words in the training corpus that share the same suffix only those that have a small frequency is a good heuristic [59] Thorsten Brants implemented these heuristics in his well-known TnT tagger and the reported accuracy for tagging ODWs is 89% [8]. A maximum likelihood estimation of a tag t given a suffix of length i $c_ic_{i-1}...c_1$ is derived from the corpus by the relation:

$$\tilde{p}(t|c_ic_{i-1}...c_1) = \frac{count_{th}(\alpha_k c_ic_{i-1}...c_1 : t)}{count_{th}(\alpha_k c_ic_{i-1}...c_1)}, \tag{5}$$

where $count_{th}(\alpha_k c_ic_{i-1}...c_1 : t)$ is the number of words ending in $c_ic_{i-1}...c_1$ which were tagged by t and their frequency is above the threshold *th*; $count_{th}$ $(\alpha_k c_ic_{i-1}...c_1)$ is total the number of words ending in $c_ic_{i-1}...c_1$with frequency above the threshold *th.* These probabilities are smoothed by successive substrings of the suffix according to the recursive formula:

$$p(t|c_ic_{i-1}...c_1) = \frac{\tilde{p}(t|c_ic_{i-1}...c_1) + \theta_i p(t|c_{i-1}...c_1)}{1+\theta_i}. \tag{6}$$

The weights θ_i may be taken context independent and set to *sample* variance (s_{N-1}) of the unconditional maximum likelihood probabilities of the tags from the training corpus:

$$s_{N-1} = \theta_i = \frac{1}{T-1}\sum_{j=1}^{T}\left(\tilde{p}(t_j) - \bar{p}\right)^2 \quad \text{and} \quad \bar{p} = \frac{1}{T}\sum_{j=1}^{T}\tilde{p}(t_j), \tag{7}$$

where T is the tagset size.

The threshold *th* for rare words was empirically set to 10 occurrences. The procedure described above can be further refined in several ways. One possibility [33] is to consider not all substrings $c_i c_{i-1} ... c_1$, but only those substrings that are real linguistic suffixes. This way, the set of possible tags for an ODW will be significantly reduced and the probability estimations will be more robust. Having a list of linguistic suffixes would also eliminate the arbitrary length of the longest "suffix" (in TnT this is 10 characters), resulting in speeding up the process.

The second problem in dealing with the data sparseness is related to estimating probabilities for n-grams w_i w_{i-1} ... w_{i-n+1} unseen in the training corpus. This problem is more general and its solution is known as smoothing We already saw a particular type of smoothing above for the ODWs processing. The basic idea is to adjust the maximum likelihood estimate of probabilities by modifying (increasing or decreasing) the actual counts for the n-grams seen in the training corpus and assigning the probability mass gained this way to unobserved n-grams. An excellent study of the smoothing techniques for language modelling is [10]. They discuss several methods among which are the following: *additive smoothing, Good-Turing Estimate, Church-Gale smoothing, Jelinek-Mercer linear interpolation* (also *Witten-Bell smoothing* and *Absolute discounting* as instantiations of the Jelinek-Mercer method), *Katz backoff* and *Kneser-Ney smoothing*. The Kneser-Ney method is generally acknowledged as the best performing smoothing technique.

The *additive smoothing* is very simple and it essentially pretends that the number of n-grams occurrences is higher with 1 than actually observed. This way, the n-grams unseen in the training corpus, but occurring in the new text will receive a small non-zero probability.

The idea of the *Good-Turing estimation* is to reallocate the probability mass of n-grams that occur r+1 times in the training data to the n-grams that occur r times with the particular case of relocating the probability mass for n-grams that were seen only once in the corpus to the n-grams that were never seen. The scheme is computing adjusted counts:

$$r^* = (r+1)\frac{n_{r+1}}{n_r}, \tag{8}$$

where n_r is the number of n-grams seen exactly r times. A more elaborated version of the adjusted counts is using ML expectations for n_{r+1} and n_r (it eliminates the risk of having discontinuous sequence of counts):

$$r^* = (r+1)\frac{E[n_{r+1}]}{E[n_r]}. \tag{9}$$

Although it is not used by itself for n-gram smoothing, the Good–Turing method is central to many smoothing techniques.

Jelinek-Mercer linear method recursively interpolates higher-order n-gram

models with lower-order n-gram models. The general formula is shown in equation (10):

$$\begin{aligned} p_{interp}(w_i|w_{i-n+1}^{i-1}) =& \lambda_{w_{i-n+1}^{i-1}} \tilde{p}(w_i|w_{i-n+1}^{i-1}) + \\ &+ (1 - \lambda_{w_{i-n+1}^{i-1}}) p_{interp}(w_i|w_{i-n+2}^{i-1}) \end{aligned} \tag{10}$$

By w_{i-n+1}^{i} is denoted the n-long sequence $w_{i-n+1}w_{i-n+2}...w_{i-1}w_i$.

Computing lambda for each n-gram is a very hard task and Jelinek and Mercer suggest clustering n-grams with similar frequency counts (see [10] for further details).

The *Katz smoothing* method uses the Good-Turing estimate and similarly to the Jelinek-Mercer method combines higher order models with lower-order models. The counts for a bi-gram with less occurrences than 5 (larger counts are considered reliable, so they are not discounted) be it r=count($w_i w_{i-1}$) are corrected according to the equations:

$$count_{Katz}(w_i w_{i-1}) = \begin{cases} d_r r & \text{if } r > 0 \\ \alpha(w_{i-1})\tilde{p}(w_i) & \text{if } r = 0 \end{cases} \tag{11}$$

$$\alpha(w_{i-1}) = \frac{1 - \sum_{w_i: count(w_i w_{i-1})>0} p_{Katz}(w_i|w_{i-1})}{1 - \sum_{w_i: count(w_i w_{i-1})>0} \tilde{p}(w_i)} \tag{12}$$

$$p_{Katz}(w_i|w_{i-1}) = \frac{count_{Katz}(w_i w_{i-1})}{\sum_{w_i} count_{Katz}(w_i w_{i-1})}. \tag{13}$$

The number of counts discounted by d_r from the global number of bigrams distribution is equal to the number of counts that should be assigned to bigrams not seen before as predicted by Good-Turing estimate (that is the number of bigrams that occurred only once). With these observations the discount ratios (for r equal to 1, 2, 3, and 4) have the expression:

$$d_r = \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \tag{14}$$

The *Kneser-Ney* smoothing technique combines, as in the Jelinek-Mercer method, the higher-level distribution with a lower-lower level distribution, with the difference that the lower-level distribution is selected so that the marginals of the higher-order distribution should match the marginals of the training data. For all interpolated probabilities they use a unique discount $D = \frac{n_1}{n_1 + 2n_2}$, where:

n_1–total number of bi-grams that occurred in the training corpus exactly once;

n_2–total number of bi-grams that occurred in the training corpus exactly twice;

Chen and Goodman [10] introduce a variant of the Kneser-Ney method which they call *Modified Kneser-Ney smoothing* and which was found having excellent

performance, outperforming the original method. Instead of using a unique discount D for all non-zero counts (as in Kneser-Ney smoothing) they use three parameters D_1, D_2 and D_{3+} that are applied to n-grams with one, two and three or more counts, respectively. The estimates for the optimum values of the parameters D_1, D_2 and D_{3+} are as follows [10]:

$$D = \frac{n_1}{n_1 + 2n_2};\ D_1 = 1 - 2D\frac{n_2}{n_1};\ D_2 = 2 - 3D\frac{n_3}{n_2};\ D_{3+} = 3 - 4D\frac{n_4}{n_3}. \quad (15)$$

4. Generative versus Discriminative Tagging Models

The model with its parameters established would allow for finding the most likely tag sequence to a new sequence of input words. In machine learning one distinguishes between generative and discriminative models [50]. A generative model is based on a model of the joint distribution $\{P_\theta(W,T) : \theta \in \Theta\}$. The best known generative model for POS tagging is the Hidden Markov Model. A discriminative model is based on a model of the conditional distribution $\{P_\theta(W|T) : \theta \in \Theta\}$. The discriminative models are also called Conditional Models (Maximum Entropy and Conditional Random Fields among others). The parameters θ are estimated by specialized methods from the training data. For tractability reasons, both generative and discriminative models make simplifying assumptions on the T sequences, but only the generative models need to make additional assumptions on W sequences. This is the main difference between the two types of models and because of this, from a theoretical point of view, the discriminative models are more powerful, although, as noticed by Toutanova [52], for the practical tagging task the improvements may not be statistically significant. Additionally, training the discriminative models is more computationally intensive.

4.1. Hidden Markov Models

Hidden Markov Models (HMM) is a generative n-gram model that allows treating the tagging of a sequence of words W (the observables) as the problem of finding the most probable (the best explanation for the observables) path traversal S (from an initial state to a final state) of a finite state system. The system's states are not directly observable. In terms of the notions introduced above, we define a first-order Hidden Markov Model (the definition is easily generalized to n-order HMMs) as a probabilistic finite-state formalism characterized by a tuple:

- HMM = $\lambda_T(\ \pi,$ A, B);
- T = finite set of states;

- π = initial state probabilities (this is the set of probabilities of the tags to be assigned to the first word of sentences; to make sense of the probabilities $p(t_1|t_o)$ the sentences are headed by a dummy word, always tagged BOS (*Begin of Sentence*) so that $p(t_1|t_o) = p(t_1|BOS) = \pi(t_1)$);
- A = transition matrix (encoding the probabilities a_{ij} to move from s_i to s_j that is, the probability $p(t_j|t_i)$ that given the tag t_i of the previous word, the current word is tagged with t_j);
- B = emission matrix (encoding the lexical probabilities $p(w_i|t_i)$, that is, the probability that being in the state s_i – tagged t_i – observe the word w_i).

HMM parameters

The parameter estimation/learning of an HMM designed for POS tagging depends on the type of available resources, along the dichotomy *supervised* vs. *unsupervised* training.

Supervised training

For supervised training a POS-corpus annotated as accurately as possible is the prerequisite resource. The minimal size of the training corpus is a very difficult question to answer because there are several factors to be taken into account: the language, the tagset size, the minimal number of occurrences of a token/tag association etc.

The set of all tags seen in the training corpus will make the tagset, based on which T parameter of the HMM is defined. All the other parameters are set by most likelihood estimations (MLE):

a) $\pi = \tilde{p}(t_i|BOS);\ \tilde{p}(t_i|BOS) = \dfrac{count(BOS, t_i)}{count(BOS)}$ that is, we count what fraction of the number of sentences have their proper first word tagged with t_i;

b) $A = \tilde{p}(t_j|t_i);\ \tilde{p}(t_j|t_i) = \dfrac{count(t_i, t_j)}{count(t_i)}$ that is, we count what fraction of the number of words tagged with t_j were preceded by words tagged with t_i;

c) $B = \tilde{p}(w_i|t_i);\ \tilde{p}(w_i|t_i) = \dfrac{count(w_i, t_i)}{count(t_i)}$ that is, we count what fraction of the number of words tagged with t_i were w_i.

It is interesting to note that knowing only the B parameter (that is a lexicon with attached probabilities for various tags of a word), then systematically choosing the highest probability tag, would ensure more than 92% and even higher accuracy for POS disambiguating texts in most of the languages. For instance, by assigning each word its highest probability tag, [36] report a tagging

accuracy as high as 94.6% (not considering ODW) on the WSJ portion of the Penn Treebank.

Unsupervised training

When an annotated training corpus is not available but a lexicon containing the ambiguity classes for the lexical stock is in place the only parameter that can be learnt from this lexicon is T.

The usual algorithm for estimating the other parameters of the HMM is Baum-Welch (BW) algorithm. From the observed sequences w_1, w_2, .. , w_N, one wishes to determine $\lambda=\{\pi, A, B\}$. The BW algorithm belongs to a family of algorithms called Expectation Maximization (EM) algorithms. They all work by using various heuristics and guessing initial parameter values when estimating the likelihood of the data under the current parameters. These likelihoods can then be used to re-estimate the parameters, iteratively until a local maximum is reached. Because of the heuristics involved, a global optimum cannot be guaranteed.

Given an observation O= w_1, w_2, ... , w_N, the BaumWelch algorithm iteratively finds $\lambda^* = \max_\lambda$ P (O|λ) - that is, the HMM λ, maximizing the probability of the observation O.

The usual independence assumptions for first-order HMM models (the statement generalizes easily for n-order HMM) are considered: the i^{th} hidden state of the model (labelling the tag for the i^{th} word) is dependent only on the previous hidden state, or: p(t_i| t_{i-1}, ...,t_1,)= p(t_i| t_{i-1}) and the i^{th} observation depends only on the i^{th} state, or: p(w_i| t_i ,... , t_1)= p(w_i| t_i).

Inference to the best tagging solution

Once the parameters of the HMM were determined (either by supervised or unsupervised training) one can proceed at solving the proper tagging problem: finding the most likely hidden transition path s_1, s_2, ..., s_N (corresponding to the sequence of tags t_1, t_2, ..., t_N) that generated the input string w_1, w_2, ..., w_N. A simple-minded solution would be to try all the possible paths (in the range of T^N) and on each state path compute p(w_j|t_i)*p(t_i|t_{i-1}). This way, we would need O(2N*T^N) computations. This number is enormous and, to get a feeling, consider a usual tagset T of 100 tags and a modest sized corpus to be tagged, containing 1000 sentences each made up on average of 20 words. The upper limit of the number of computations for tagging this corpus would be $2 * 1,000 * 20 * 100^{20} = 4 * 10^{44}$. Fortunately, there are many ways to make the computation tractable.

The best known algorithm to solve this problem is Viterbi's which is demonstrated[1] to find the optimal sequence of hidden states doing at most N*T^2 computations. To take the previous example, Viterbi's algorithm would need no more than $2 * 10^8$ computations, that is, $2 * 10^{36}$ less! The algorithm looks at each state j at time t which could emit the word w_t (that is for which $b_j(w_t)$

[1]See for instance `https://onlinecourses.science.psu.edu/stat857/node/203`

is non-null) and for all the transitions that lead into that state, it decides which of them, say i, was the most likely to occur, *i.e.* the transition with the greatest accumulated score $\gamma_{t-1}(i)$. The state i from which originated the transition with the highest accumulated score is stored as a back pointer to the current state $j$and it is assigned the accumulated score $\gamma_t(j) = b_j(w_t) * \gamma_{t-1}(i)$. When the algorithm reaches the end of the sentence (time t = N), it determines the final state as before and computes the Viterbi path by following the back pointers (backtracking). The probability of this path is the accumulated score of the final state.

4.2. Discriminative Models

This is a class of models that were defined for solving problems where the two fundamental hypotheses used by the generative models were too weak or grossly inapplicable. The conditional models also consider more realistically the inherent lack of full data sets required to build a robust and wide coverage statistical model. The joint probability distributions are replaced with conditional probabilities distributions where conditioning is restricted to available data and they are enabled to consider not only the identity of observables (word-forms) but many other relevant properties they may have such as prefixes, suffixes, embedded hyphens, starting with an uppercase letter, etc. Dependencies among non-adjacent input words can also be taken into account.

Among the conditional models, the most successful for the POS tagging are Maximum Entropy (ME) Models (including Maximum Entropy Markov Models) and the Conditional Random Fields (CRF) models (with different variants *Linear-Chain CRF* and *Skip-Chain CRF*).

Maximum Entropy Models

The Maximum Entropy language modelling was first used by [20]. The Maximum Entropy (ME) based taggers are among the best performing since they take into account "diverse forms of contextual information in a principled manner, and do not impose any distributional assumptions on the training data [45]. If one denoted by H the set of contextual hints (or *histories*) for predicting tags and by T the set of possible tags, then one defines the event space as $E \subseteq H \otimes T$. The common way to represent contextual information is by means of binary valued features (constraint functions) f_i on event space f_i: $E \rightarrow \{0,1\}$, subject to a set of restrictions. The model requires that the expected value of each feature f_i according to the model p should match the observed value and, moreover, it should be a constant:

$$C_i = \sum_{\{(h,t)\}} p(h,t) f_i(h,t). \tag{16}$$

Computing this expected value according to p requires summing over all events (h, t) which is not practically possible. The standard approximation

[16], [46] is to use the observed relative frequencies of the events with the simplification that this is zero for unseen events:

$$\tilde{p}(h,t) = \sum_{\{(h,t)\}} \tilde{p}(h)p(t|h)f_i(h,t).$$

It may be shown (see [45]) that the probability of tag t given the context h for a ME probability distribution has the form:

$$p(t|h) = \pi(h) \prod_{j=1}^{k} \alpha_j^{f_j(h,t)}, \tag{17}$$

where $\pi(h) = \dfrac{1}{\sum_t \prod_{i=1}^{k} \alpha_i^{f_j(h,t)}}$ is a normalisation constant, $\{\alpha_1, \ ... \ , \alpha_k\}$ are the positive model parameters and $\{f_1, \ ... \ , f_k\}$ are the binary features used to contextually predict the tag t. The history h_i is defined as follows: h_i=$\{w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2}\}$ with w_i the current word for which the tag is to be predicted, w_{i-1} , w_{i-2} , t_{i-1} and t_{i-2} the preceding two words and their respective predicted tags and w_{i+1} , w_{i+2} the following two words. For a given history *h*and a considered tag *t*for the current word w_i, a feature $f_i(h,t)$ may refer to any word or tag in the history and should encode information that might help predict t.

ME parameters

The parameters $\{\alpha_1, \ ... \ , \alpha_k\}$ act as weights for each active feature which together contribute to predicting a tag t_i for w word w_i. They are chosen to maximize the likelihood of the training data (a tagged corpus of N words):

$$L(p) = \prod_{i=1}^{N} \pi \prod_{j=1}^{k} \alpha_j^{f_j(h,t)}. \tag{18}$$

The algorithm for finding the parameters of the distribution that uniquely maximizes the likelihood L(p) over distributions of the form (17) that satisfy the constraints specified by the features is called *Generalized Iterative Scaling* – GIS.

Inference to the best tagging solution

This is basically a beam search algorithm (the beam size is a parameter; the larger the beam, the longer the tagging time is; Radnaparkhi [45] recommends the value of 5). The algorithm[2] finds the MLE tag sequence t_1 ... t_N for the input string:

$$\max P(t_1 ... t_N | w_1 ... w_N) = \prod_{i=1}^{N} p(t_i|h_i). \tag{19}$$

[2]See [45] for a complete description of the algorithm.

There are various available ME taggers with tagging accuracy ranging between 97% and 98% (MaxEnt, Stanford tagger, NLTK tagger, to name just a few).

The popularity of the ME taggers is due to the wide range of context dependencies that may be encoded via the features. Nevertheless, as has already been noticed by other researchers [46], selection of the features is very important and this is a matter of human expertise (same training data but different features would more often than not lead to different performances). On top of this, most of the best discriminating features are language dependent. While the GIS algorithm is guaranteed to converge (provided the constraints are consistent) there is no theoretical bound on the number of iterations required and, correlated with the intensive computation, one may decide to stop the iterations before reaching the optimal solution.

Conditional Random Field Model

The Conditional Random Field (CRF) model was introduced in [35]. It is a very general discriminative model that uses the exponential conditional distribution, similar to the maximum entropy model. As in ME models, binary features are used as triggers for contextual information that might support a prediction.

The generalization consists of giving to any feature access to the entire sequence of observables, so that it can be activated to support prediction t_i for the observable w_i, taking into account whatever attribute is needed from the sequence w_1, w_2, ..., w_N. The tagging problem has been addressed with a particular form of the model, called Linear Chain Conditional Random Field, which combines the advantages of discriminative modelling (feature based) and sequence modelling (order based). The major motivation for CRF was to deal with one weakness of discriminative models based on finite states, namely the *label bias problem.* This problem is related to the computing of the transition probabilities which, ignoring the observables, are biased towards states which have fewer outgoing transitions. Unlike previous non-generative finite state models, which use per state exponential models for the conditional probabilities of next state given the current state, a CRF has a single exponential model for the probability of the entire sequence of states given the observation sequence [35]. CRF uses a normalization function which is not a constant but a function of the observed input string.

The Skip-Chain CRF is a Linear-Chain CRF with additional long-distance edges (*skip edges*) between related words. The features on skip edges may incorporate information from the context of both endpoints, so that strong evidence at one endpoint can influence the label at the other endpoint [35]. The relatedness of the words connected by a skip edge may be judged based on orthography similarity (*e.g.* Levenshtein distance), or semantic similarity (*e.g.* WordNet-based synonymy or hyponymy).

The procedure for the estimation of CRF model parameters is an improved

version of the GIS, due to [21]. The inference procedure is a modified version of the Viterbi algorithm (see [35] for details).

5. Tagsets design

While corpus structuring has been a long-standing topic in corpus linguistics [34], [49], [39], [3], [31] and many others, it is surprising how little effort has been devoted to designing appropriate tagsets for different languages and different processing approaches. We do not consider here work on automatic induction of tagsets [14], [4], [26], etc., but discuss the expert design of morpho-lexical tagsets.

It is part of corpus linguistics lore that in order to get high accuracy in statistical POS disambiguation, one needs small tagsets and reasonably large training data. The effect of tagset size on tagger performance has been discussed in [23]. The larger the tagset, the larger the necessary training corpora is [2]. There is a dependency between the size of the training data and the size of the tagset which unfortunately is not linear. This is understandable given that words in particular parts of speech are distributed unevenly in corpora, and each word and its associated POS tag has to be seen a minimal number of times to extract reliable statistics.

In the past, most of the work in POS tagging focussed on the performance improvement of the learners and taggers [1], [18], [40], [22], [27], etc., relying on historical tagsets developed for English. The first concerted research effort directed at addressing the language specificity of tagsets with an eye for multilinguality and interoperability criteria was carried out within the European projects EAGLES [51], [41] Multext (`aune.lpl.univ-aix.fr/projects/multext/`), and its spin-off Multext-East [24]. These tagsets proved to be descriptively adequate for all European languages and were eventually used for encoding lexica in more than 25 languages. The multilingual lexical tagset specifications developed in these projects required in the case of some languages (especially Slavic languages) several thousands of morpho-lexical descriptors (MSD). To overcome the data sparseness problem, [53] introduced a technique called tiered tagging in which a reduced tagset (C-tags), induced from the lexical tagset (MSDs), is used to produce an initial tagging followed by an information recovery process that replaces the C-tags with appropriate MSDs. The tiered tagging methodology is independent on the tagging model. There are several implementations available for both generative and discriminative tagging models described in this paper. When the reduced tagset is generated without information loss [7] the recovering process is strictly deterministic. In [56] it is described an algorithm that has been used to generate optimal reduced tagsets without information loss (baseline tagsets) from the respective MSD tagsets for five Central Eastern European languages (Romanian, Slovene, Hungarian, Czech and Estonian) plus English. The tagsets reduction ratios were ranging from 3 (English) to 11 (Romanian). Very good results were also reported for German [30] and Polish [44].

When the baseline tagsets are further shrunk, the information loss is inherent and the recovering process needs additional knowledge sources. In the initial proposal of the tiered tagging method, the recovering process was rule-based and language specific. For Romanian, [53] used 18 regular expression rules. Depending on the specific case of recovery ambiguity, these rules inspect left, right, or both contexts within a limited distance (the maximum span is 4). Besides the linguistic expertise required for rule writing, the rule-based recovery phase of tiered tagging works only for the word-forms the ambiguity classes of which are known to the tagger. These limitations were eliminated in a follow-up implementation of the tiered tagging method, called METT [9]. METT is a tiered tagging system that uses the maximum entropy (ME) approach to automatically induce the contextual mappings between the C-tagset and the MSD-tagset. This method requires a training corpus tagged twice: the first time with MSDs and the second time with C-tags. Transforming an MSD-annotated corpus into its proper C-tagset variant can be carried out deterministically. METT learns non-lexicalized probabilistic mappings from C-tagset to MSD-tagset. Therefore, it is able to assign a contextually adequate MSD to a C-tag labelling an out-of-dictionary word and even to correct wrongly initially assigned C-tags. The tiered tagging methodologies implemented by METT [9] and TTL [33] have been turned into SOAP-based web-services [57] and into REST-based web-services [32] running for "en", "ro", "fr" and "de" languages.

There are several applications for which identification of only the part of speech of a token (without any other attribute value) is sufficient. For such applications the desired tagset would contain about a dozen tags (most common morpho-lexical specifications distinguish 13-14 grammar categories). This situation is the opposite of the one discussed above, where there exist very large tagsets). Is the C-tagset optimality issue relevant for such a shallow tagset? In [55] it is described an experiment which gives a positive answer: the Gold Standard training corpus and 15 test sets annotated with MSDs were modified by replacing all the lexical tags with the corresponding grammar category (position 0 in the Multext-East linear encoding). Thus, the tagset in the training corpus and the test sets was reduced to 14 tags. The tagger was trained on this new "Gold Standard" and evaluated on the test sets. The average tagging accuracy was never higher than 93%. When the same texts were tagged with the optimal C-tagset, and after removal of all the C-tag attributes, the tagging accuracy was never below 99% (average accuracy 99.35%).

6. Conclusions

In this paper we addressed the problem of part-of-speech tagging and its most popular approaches. We briefly discussed the data sparseness problem and the most effective methods to limit the inherent lack of sufficient training data for statistical modelling. The average performance of the state-of-the-art taggers for most of the languages is 97-98%. This figure might sound impres-

sive, yet if we consider an average sentence length of 30 words it means that on average every third sentence (due to local dependencies, tagging errors tend to cluster in the same sentences) may contain 2-3 tagging errors which could be harmful for its higher level processing (syntactic, semantic, discourse). With a limited number of ambiguities (k-best tagging) left in the output, for subsequent linguistically more informed processors, the accuracy of the POS-tagging could be almost 100% accurate. The multilinguality and interoperability requirements for the modern tagging technology as well as the availability of more lexical resources led to larger tagsets than used earlier and, consequently, the appropriate design of the tagsets becomes an important activity. The maximally informative linguistic encodings found in standardized computational lexicons for many languages are too numerous to be directly used as POS tagsets. The tiered tagging methodology is one way to cope with large lexical tagsets in POS tagging and still ensure robustness and high accuracy for the tagging process.

References

[1] BAAYEN H., SPROAT R., *Estimating Lexical Priors for Low-Frequency Morphologically Ambiguous Forms*. Computational Linguistics, 22(2), 1996, 155–166.

[2] BERGER A. L., DELLA PIETRA S, A., DELLA PIETRA V.J., *A Maximum Entropy Approach to Natural Language Processing*. Computational Linguistics, 22(1), 1996, 39–72.

[3] BIBER D., CONRAD S., REPPEN R., *Corpus Linguistics, Investigating Language Structure and Use*, Cambridge: Cambridge UP, 1998.

[4] BIEMANN C., *Unsupervised part-of-speech tagging employing efficient graph clustering*. In Proceedings of COLING ACL 2006, Morristown, NJ, USA, 2006, 7–12.

[5] BLACK E., JELINEK F., LAFFERTY, J., MERCER, R., ROUKOS S., *Decision Tree Models Applied to the Labeling of Text with Parts-of-Speech*. In Proceedings of the HLT'91 Workshop on Speech and Natural Language, Harriman, New York, 1992, 117–121.

[6] BOROŞ T., ION R., TUFIŞ D., *Large tagset labelling using Feed Forward Neural Networks. Case study on Romanian Language*, ACL 2013, Sofia, Bulgaria 5-8 August, 2013.

[7] BRANTS T. *Tagset Reduction without Information Loss*. In Proceedings of the 33^{rd} Annual Meeting of the ACL. Cambridge, MA, 1995, 287–289.

[8] BRANTS T. *TnT – A Statistical Part-of-Speech Tagger*. In Proceedings of the 6^{th} Applied NLP Conference. Seattle, WA, 2000, 224–231.

[9] CEAUŞU A., *Maximum Entropy Tiered Tagging*. In Proceedings of the Eleventh ESSLLI Student Session, ESSLLI, Malaga, Spain, 2006, 173–179.

[10] CHEN S. F., GOODMAN J., *An Empirical Study of Smoothing Techniques for Language Modelling*. Technical Report TR-10-98, August, Computer Science Group at Harvard University, Cambridge, Massachusetts, 1998.

[11] CHRISTODOULOPOULOS C., GOLDWATER S., STEEDMAN M., *Two Decades of Unsupervised POS Induction: How far have we come?* In Proceedings of the 2010 EMNLP Conference, MIT Massachusetts, 2010, 575–584.

[12] CHURCH K., *A stochastic parts program and noun phrase for unrestricted text.* In Proceedings of IEEE 1989 International Conference on Acoustics, Speech and Signal Processing, Glasgow, 1989, 695–698.

[13] CHURCH K., *Current Practice in Part of Speech Tagging and Suggestions for the Future.* In Simmons (ed.), Abornik praci: In Honor of Henry Kucera, Michigan Slavic Studies, 1992, 13-48.

[14] CLARK A., *Combining distributional and morphological information for part of speech induction.* In Proceedings of EACL 2003, Budapest, Hungary, 2003, 59–66.

[15] COLLINS M., *Discriminative training methods for hidden Markov models: Theory and experiments with the perceptron algorithm.* In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Philadelphia, Pennsylvania, 2002, 1–8.

[16] CURRAN J. R., CLARK S., *Investigating GIS and Smoothing for Maximum Entropy Taggers.* In Proceedings of the EACL 2003, Budapest, Hungary, 2003, 91–98.

[17] CUSSENS J., *Part-of-speech tagging using Progol. Inductive Logic Programming.* LNCS Volume 1297, Springer Berlin/Heildelberg, 1997, 93–108.

[18] CUTTING D., KUPIEK J., PEDERSEN J., SIBUN P., *A Practical Part-of-Speech Tagger.* In Proceedings of Third Conference on Applied Language Processing, Trento, Italy, 1992, 133–140.

[19] DAELEMANS W., ZAVREL J., BERCK P., GILLIS S., *MBT: A Memory-Based Part-of-Speech Tagger Generator.* In Proceedings of 4^{th} Workshop on Very Large Corpora, Copenhagen, Denmark, 1996, 14–27.

[20] DELLA PIETRA, S. A., DELLA PIETRA V. J., MERCER R., ROUKOS S., *Adaptive Language Modeling Using Minimum Discriminant Estimation.* In Proceedings of the International Conference on Acoustics, Speech and Signal Processing, San Francisco, 1992, 633–636.

[21] DELLA PIETRA, S. A., DELLA PIETRA V. J., LAFFERTY J., *Inducing features of random fields.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(4), 1997, 380–393.

[22] DERMATAS E., KOKKINAKIS G., *Automatic Stochastic Tagging of Natural Language Texts.* Computational Linguistics, 21(2), 1995, 321–350.

[23] ELWORTHY D. *Tagset Design and Inflected Languages.* In Proceedings of the ACL SIGDAT Workshop, Dublin, 1995.

[24] ERJAVEC T., *MULTEXT-East XML: an investigation of a schema for language engineering and corpus linguistics.* In Tufiş D. and Forăscu C. (eds.) Multilinguality and Interoperability in Language Processing with Emphasis on Romanian, Romanian Academy Publishing House, 2010, 15–38.

[25] GALE W. A., SAMPSON G., *Good-Turing Frequency Estimation without Tears.* In Journal of Quantitative Linguistics, 2/3, 1995, 217–237.

[26] HAGHIGHI A., KLEIN D., *Prototype-Driven Learning for Sequence Models.* In Proceedings of NAACL 2006, Morristown, NJ, USA, 2006, 320–327.

[27] HAJIČ J., *Morphological Tagging: Data vs. Dictionaries.* In Proceedings of the ANLP/NAACL, Seattle, 2000, 94–101.

[28] HALTEREN H.V. (ed.). *Syntactic Wordclass Tagging*. Text, Speech and Language book series, vol. 9, Kluwer Academic Publishers, Dordrecht/Boston/ London, 1999.

[29] HERBST E., JOACHIMS T., *SVM^{hmm} Sequence Tagging with Structural Support Vector Machines and its Application to Part-of-Speech Tagging*, `http://www.cs.cornell.edu/people/tj/svm_light/old/svm_hmm_v2.13.html`, 2007.

[30] HINRICHS E., TRUSHKINA J., *Forging Agreement: Morphological Disambiguation of Noun Phrases*. In Proceedings of the Workshop Treebanks and Linguistic Theories, Sozopol, Bulgaria, 2002, 78–95.

[31] IDE NANCY, *Encoding Linguistic Corpora*. In Proceedings of the Sixth Workshop on Very Large Corpora, 1998, 9–17.

[32] ION R., CEAUŞU A., ŞTEFĂNESCU D., TUFIŞ D., *Interoperable and Multilingual Web services*. In Iftene, A., Teodorescu H. N., Cristea D., Tufiş D. (eds.) Language resources and tools for processing Romanian (in Romanian). "A. I. Cuza" Univ. Publ. House, 2010, 167–176.

[33] ION R., *Automatic Semantic Disambiguation Methods. Applications for English and Romanian* (in Romanian). PhD Thesis, Romanian Academy, 2007.

[34] JOHANSSON S., ATWELL E., GARSIDE R., LEECH G., *The Tagged LOB Corpus Users' Manual*. Norwegian Computing Centre for Humanities, Bergen, Norway, 1986, 149 pag., `http://khnt.hit.uib.no/icame/manuals/lobman/`

[35] LAFFERTY J., MCCALLUM A., PEREIRA F. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. In Proceedings of the 18^{th} International Conference in Machine Learning (ICML-2001), Berkshires, Mass, USA, 2001, 282–289.

[36] LEE K. Y., HAGHIGI A., BARZILAY R., *Simple Type-Level Unsupervised POS Tagging*. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, MIT, Massachusetts, 2010, 853–861.

[37] LEECH G., SMITH N., *POS-tagging Error Rates*. BNC2 POS tagging Manual, UCREL, Lancaster University, 2000, `http://www.natcorp.ox.ac.uk/docs/bnc2error.htm`

[38] MARAGOUDAKIS M., KERMANIDIS K., FAKOTAKIS N., *Towards a Bayesian Stochastic Part-Of-Speech and Case Tagger of Natural Language Corpora*. In Proceedings of the Conference on Corpus Linguistics, March, Lancaster, UK, 2003.

[39] MARCUS M. P., SANTORINI B., MARCINKIEWICZ M. A., *Building a Large Annotated Corpus of English: The Penn Treebank*. Computational Linguistics, 19(2), 1993, 313–330.

[40] MERIALDO B., *Tagging English Text with a Probabilistic Tagger*. Computational Linguistics, 20(2), 1994, 155–172.

[41] MONACHINI M., CALZOLARI N. (eds.), *Synopsis and Comparison of Morphosyntactic Phenomena Encoded in Lexicons and Corpora*, 1996, `http://www.ilc.cnr.it/EAGLES96/morphsyn/morphsyn.html`

[42] PADRÓ L., MÁRQUEZ L., *On the Evaluation and Comparison of Taggers: the Effect of Noise in Testing Corpora*. In Proceedings of COLING-ACL'98.Montreal, Canada, 1998, 997–1002.

[43] RABINER L. R., *A tutorial on Hidden Markov Models and selected applications in speech recognition*. Proceedings of the IEEE. Volume 77, Issue 2, 1989, 257–286.

[44] RADZISZEWSKI A. *A Tiered CRF Tagger for Polish*. In Intelligent Tools for Building a Scientific Information Platform, SCI 467, 2013, 215-230, DOI: 10.1007/978-3-642-35647-6_16

[45] RATHAPARKHI A., *A Maximum Entropy Part of Speech Tagger*. In Proceedings of EMNLP'96, Philadelphia, Pennsylvania, 1996, 133–142.

[46] ROSENFELD R., *A Maximum Entropy Approach to Statistical Language Modelling*. Computer Speech and Language, Volume 10, 1996, 187–228.

[47] SAMUELSSON C., VUOTILAINEN A., *Comparing a Linguistic and a Stochastic Tagger*. In Proceedings of Joint EACL/ACL Conference, Madrid, Spain, 1997, 241–253.

[48] SCHMID H., *Probabilistic Part-of-Speech Tagging Using Decision Trees*. In Proceedings of International Conference on New Methods in Language Processing, Manchester, UK, 1994, `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`

[49] SINCLAIR, J., *The automatic analysis of corpora*. In Svartvik, J. (ed.) Directions in Corpus Linguistics (Proceedings of Nobel Symposium 82). Berlin: Mouton de Gruyter, 1992, 379–397.

[50] SUTTON C., MCCALLUM A., *An Introduction to Conditional Random Fields for Relational Learning*. In (Getoor L., Taskar B. eds). Introduction to Statistical Relation Learning. MIT Press, 2006, 93–128.

[51] TEUFEL S., SCHILLER A., HEID U., *Task on Tagset and Tagger Interaction*. EAGLES Validation (WP-4) Final Report, 1996, 53 pag.

[52] TOUTANOVA K., *Competitive generative models with structure learning for NLP classification tasks*. In Proceedings of EMNLP, Stanford, 2006, 576–584.

[53] TUFIŞ D., *Tiered Tagging and Combined Classifiers*. In F. Jelinek, E. Nöth (eds) Text, Speech and Dialogue, Lecture Notes in Artificial Intelligence 1692, Springer, 1999, 28–33.

[54] TUFIŞ D., MASON O., *Tagging Romanian Texts: a Case Study for QTAG, a Language Independent Probabilistic Tagger*. In Proceedings of the International Conference on Language Resources and Evaluation, Granada, Spain, 1998, 589–596.

[55] TUFIŞ D., *Using a Large Set of EAGLES-compliant Morpho-Syntactic Descriptors as a Tagset for Probabilistic Tagging*. In Proceedings of the 3^{rd} International Conference on Language Resources and Evaluation, Athens, Greece, 2000, 1105–1112.

[56] TUFIŞ D., DRAGOMIRESCU L., *Tiered Tagging Revisited*. In Proceedings of the 4^{th} International Conference on Language Resources and Evaluation, Lisbon, Portugal, 2004, 39–42.

[57] TUFIŞ D., ION R., CEAUŞU A., ŞTEFĂNESCU D., *RACAI's Linguistic Web Services*. In Proceedings of the 6^{th} Language Resources and Evaluation Conference, Marrakech, Morocco, 2008, 327–333.

[58] VITERBI A. J., *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm*. IEEE Transactions on Information Theory 13 (2), 1967, 260–269.

[59] WEISCHEDEL R., METEER M., SCHWARTZ R., RAMSHAW L., PALMUCCI J., *Coping with Ambiguity and Unknown Words through Probabilistic Models*. Computational Linguistics, 19(2), 1993, 219–242.