Hidden Markov tree models for semantic class induction

Edouard Grave, Guillaume Obozinski, Francis Bach

Hidden Markov tree models for semantic class induction

Édouard Grave
Inria - Sierra Project-Team
École Normale Supérieure
Paris, France
Edouard.Grave@inria.fr

Guillaume Obozinski
Université Paris-Est, LIGM
École des Ponts - ParisTech
Marne-la-Vallée, France
Guillaume.Obozinski@imagine.enpc.fr

Francis Bach
Inria - Sierra Project-Team
École Normale Supérieure
Paris, France
Francis.Bach@ens.fr

Abstract

In this paper, we propose a new method for semantic class induction. First, we introduce a generative model of sentences, based on dependency trees and which takes into account homonymy. Our model can thus be seen as a generalization of Brown clustering. Second, we describe an efficient algorithm to perform inference and learning in this model. Third, we apply our proposed method on two large datasets (10^8 tokens, 10^5 words types), and demonstrate that classes induced by our algorithm improve performance over Brown clustering on the task of semi-supervised supersense tagging and named entity recognition.

1 Introduction

Most competitive learning methods for computational linguistics are supervised, and thus require labeled examples, which are expensive to obtain. Moreover, those techniques suffer from data scarcity: many words only appear a small number of time, or even not at all, in the training data. It thus helps a lot to first learn word clusters on a large amount of unlabeled data, which are cheap to obtain, and then to use this clusters as features for the supervised task. This scheme has proven to be effective for various tasks such as named entity recognition (Freitag, 2004; Miller et al., 2004; Liang, 2005; Faruqui et al., 2010), syntactic chunking (Turian et al., 2010) or syntactic dependency parsing (Koo et al., 2008; Haffari et al., 2011; Tratz and Hovy, 2011). It was also successfully applied for transfer learning of multilingual structure by Täckström et al. (2012).

The most commonly used clustering method for semi-supervised learning is the one proposed by Brown et al. (1992), and known as Brown clustering. While still being one of the most efficient word representation method (Turian et al., 2010), Brown clustering has two limitations we want to address in this work. First, since it is a hard clustering method, homonymy is ignored. Second, it does not take into account syntactic relations between words, which seems crucial to induce semantic classes. Our goal is thus to propose a method for semantic class induction which takes into account both syntax and homonymy, and then to study their effects on semantic class learning.

In this paper, we start by introducing a new unsupervised method for semantic classes induction. This is achieved by defining a generative model of sentences with latent variables, which aims at capturing semantic roles of words. We require our method to be scalable, in order to learn models on large datasets containing tens of millions of sentences. More precisely, we make the following contributions:

- We introduce a generative model of sentences, based on dependency trees, which can be seen as a generalization of Brown clustering,
- We describe a fast approximate inference algorithm, based on message passing and online EM for scaling to large datasets. It allowed us to learn models with 512 latent states on a dataset with hundreds of millions of tokens in less than two days on a single core,
- We learn models on two datasets, Wikipedia articles about musicians and the NYT corpus,

and evaluate them on two semi-supervised tasks, namely supersense tagging and named entity recognition.

1.1 Related work

Brown clustering (Brown et al., 1992) is the most commonly used method for word cluster induction for semi-supervised learning. The goal of this algorithm is to discover a clustering function $\mathcal{C}$ from words to clusters which maximizes the likelihood of the data, assuming the following sequential model of sentences:

$$\prod_k p(w_k \mid \mathcal{C}(w_k))p(\mathcal{C}(w_k) \mid \mathcal{C}(w_{k-1})).$$

It can be shown that the best clustering is actually maximizing the mutual information between adjacent clusters. A greedy agglomerative algorithm was proposed by Brown et al. (1992) in order to find the clustering $\mathcal{C}$, while Clark (2003) proposed to use the exchange clustering algorithm (Kneser and Ney, 1993) to maximize the previous likelihood. One of the limitations of this model is the fact that it neither takes into account homonymy or syntax.

Another limitation of this method is the complexity of the algorithms proposed to find the best clustering. This led Uszkoreit and Brants (2008) to consider a slightly different model, where the class-to-class transitions are replaced by word-to-class transitions:

$$\prod_k p(w_k \mid \mathcal{C}(w_k))p(\mathcal{C}(w_k) \mid w_{k-1}).$$

Thanks to that modification, Uszkoreit and Brants (2008) designed an efficient variant of the exchange algorithm, allowing them to train models on very large datasets. This model was then extended to the multilingual setting by Täckström et al. (2012).

Semantic space models are another family of methods, besides clustering, that can be used as features for semi-supervised learning. In those techniques, words are represented as vectors in a high-dimensional space. These vectors are obtained by representing the unlabeled corpus as a word-document co-occurrence matrix in the case of *latent semantic analysis* (LSA) (Deerwester et al., 1990), or word-word co-occurrence matrix in the case of the *hyperspace analog to language* model (HAL) (Lund and Burgess, 1996). Dimension reduction is then performed, by taking the singular value decomposition of the co-occurrence matrix, in order to obtained the so-called semantic space. Hofmann (1999) proposed a variant of LSA, which corresponds to a generative model of document. More recently, Dhillon et al. (2011) proposed a method based on canonical correlation analysis to obtained a such word embeddings.

A last approach to word representation is latent Dirichlet allocation (LDA), proposed by Blei et al. (2003). LDA is a generative model where each document is viewed as a mixture of topics. The major difference between LDA and our model is the fact that LDA treats documents as bags of words, while we introduce a model of sentences, taking into account the syntax. Griffiths et al. (2005) defined a composite model, using LDA for topic modeling and an HMM for syntax modeling. This model, HMM-LDA, was used by Li and McCallum (2005) for semi-supervised learning and applied to part-of-speech tagging and Chinese word segmentation. Séaghdha (2010) proposed to use topic models, such as LDA, to perform selectional preference induction.

Finally, Boyd-Graber and Blei (2009) proposed a variant of LDA, using parse trees to include the syntax. Given that we aim for our classes to capture as much of the word semantics reflected by the syntax, such as the semantic roles of words, we believe that it is not necessarily useful or even desirable that the latent variables should be determined, even in part, by topic parameters that are sharing information at the document level. Moreover, our model being significantly simpler, we were able to design fast and efficient algorithms, making it possible to use our model on much larger datasets, and with many more latent classes.

2 Model

In this section, we introduce our probabilistic generative model of sentences. We start by setting up some notations. A sentence is represented by a K-tuple $\mathbf{w} = (w_1, ..., w_K)$ where each $w_k \in \{1, ..., V\}$ is an integer representing a word and V is the size of the vocabulary. Our goal will be to infer a K-tuple $\mathbf{c} = (c_1, ..., c_K)$ of semantic classes, where each $c_k \in \{1, ..., C\}$ is an integer representing a semantic class, corresponding to the word w_k.

The generation of a sentence can be decomposed in two steps: first, we generate the semantic classes according to a Markov process, and

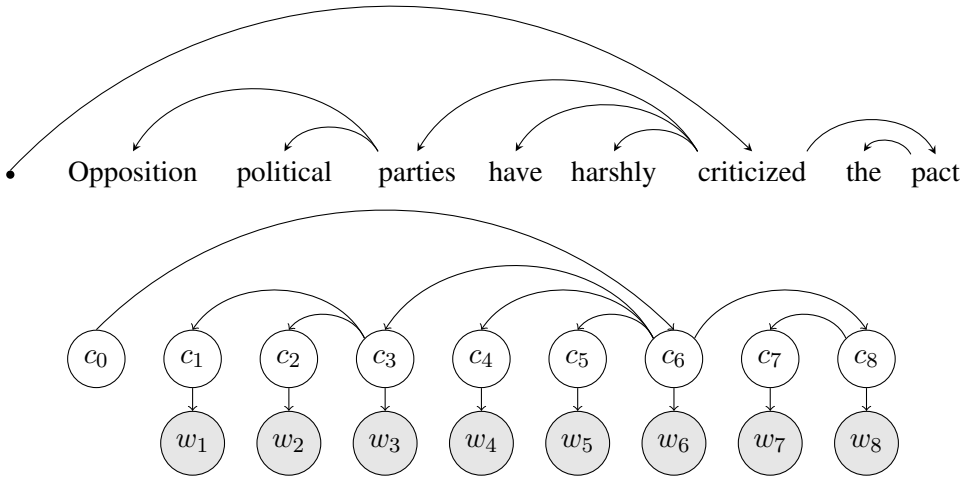

Figure 1: Example of a dependency tree and its corresponding graphical model.

then, given each class c_k, we generate the corresponding word w_k independently of other words. The Markov process used to generate the semantic classes will take into account selectional preference. Since we want to model homonymy, each word can be generated by multiple classes.

We now describe the Markov process we propose to generate the semantic classes. We assume that we are given a directed tree defined by the function $\pi : \{1, ..., K\} \mapsto \{0, ..., K\}$, where $\pi(k)$ represents the unique parent of the node k and 0 is the root of the tree. Each node, except the root, corresponds to a word of the sentence. First, we generate the semantic class corresponding to the root of the tree and then generate recursively the class for the other nodes. The classes are conditionally independent given the classes of their parents. Using the language of probabilistic graphical models, this means that the distribution of the semantic classes factorizes in the tree defined by π (See Fig. 1 for an example). We obtain the following distribution on pairs $(\mathbf{w}, \mathbf{c})$ of words and semantic classes:

$$p(\mathbf{w}, \mathbf{c}) = \prod_{k=1}^{K} p(c_k \mid c_{\pi(k)}) p(w_k \mid c_k),$$

with c_0 being equal to a special symbol denoting the root of the tree.

In order to fully define our model, we now need to specify the observation probability distribution $p(w_k \mid c_k)$ of a word given the corresponding class and the transition probability distribution $p(c_k \mid c_{\pi(k)})$ of a class given the class of the parent. Both these distributions will be categorical (and thus multinomial with one trial). The corresponding parameters will be represented by the stochastic matrices $\mathbf{O}$ and $\mathbf{T}$ (i.e. matrices with non-negative elements and unit-sum columns):

$$p(w_k = i \mid c_k = j) = O_{ij},$$
$$p(c_k = i \mid c_{\pi(k)} = j) = T_{ij}.$$

Finally, we introduce the trees that we consider to define the distribution on semantic classes. (We recall that the trees are assumed given, and not a part of the model.)

2.1 Markov chain model

The simplest structure we consider on the semantic classes is a Markov chain. In this special case, our model reduces to a hidden Markov model. Each semantic class only depends on the class of the previous word in the sentence, thus failing to capture selectional preference of semantic class. But because of its simplicity, it may be more robust, and does not rely on external tools. It can be seen as a generalization of the Brown clustering algorithm (Brown et al., 1992) taking into account homonymy.

2.2 Dependency tree model

The second kind of structure we consider to model interactions between semantic classes is a syntactic dependency tree corresponding to the sentence. A dependency tree is a labeled tree in which nodes correspond to the words of a sentence, and edges represent the grammatical relations between those words, such as *nominal subject*, *direct object* or *determiner*. We use the Stanford typed dependencies basic representations, which always form a tree (De Marneffe and Manning, 2008).

We believe that a dependency tree is a better structure than a Markov chain to learn semantic classes, with no additional cost for inference and learning compared to a chain. First, syntactic dependencies can capture long distance interactions between words. See Fig. 1 and the dependency between `parties` and `criticized` for an example. Second, the syntax is important to model selectional preference. Third, we believe that syntactic trees could help much for languages which do not have a strict word order, such as Czech, Finnish, or Russian. One drawback of this model is that all the children of a particular node share the same transition probability distribution. While this is not a big issue for nouns, it is a bigger concern for verbs: subject and object should not share the same transition probability distribution.

A potential solution would be to introduce a different transition probability distribution for each type of dependency. This possibility will be explored in future work.

2.3 Brown clustering on dependency trees

As for Brown clustering, we can assume that words are generated by a single class. In that case, our model reduces to finding a deterministic clustering function $\mathcal{C}$ which maximizes the following likelihood:

$$\prod_k p(w_k \mid \mathcal{C}(w_k)) p(\mathcal{C}(w_k) \mid \mathcal{C}(w_{\pi(k)})).$$

In that case, we can use the algorithm proposed by Brown et al. (1992) to greedily maximize the likelihood of the data. This model can be seen as a generalization of Brown clustering taking into account the syntactic relations between words.

3 Inference and learning

In this section, we present the approach used to perform learning and inference in our model. Our goal here is to have efficient algorithms, in order to apply our model to large datasets (10^8 tokens, 10^5 words types). The parameters $\mathbf{T}$ and $\mathbf{O}$ of the model will be estimated with the maximum likelihood estimator:

$$\hat{\mathbf{T}}, \hat{\mathbf{O}} = \arg\max_{\mathbf{T},\mathbf{O}} \prod_{n=1}^{N} p(\mathbf{w}^{(n)} \mid \mathbf{T}, \mathbf{O}),$$

where $(\mathbf{w}^{(n)})_{n \in \{1,\dots,N\}}$ represents our training set of N sentences.

First, we present an online variant of the well-known expectation-maximization (EM) algorithm, proposed by Cappé and Moulines (2009), allowing our method to be scalable in term of numbers of examples. Then, we present an approximate message passing algorithm which has a linear complexity in the number of classes, instead of the quadratic complexity of the exact inference algorithm. Finally, we describe a state-splitting strategy to speed up the learning.

3.1 Online EM

In the batch EM algorithm, the E-step consists in computing the expected sufficient statistics τ and ω of the model, sometimes referred as pseudo-counts, corresponding respectively to $\mathbf{T}$ and $\mathbf{O}$:

$$\tau_{ij} = \sum_{n=1}^{N} \sum_{k=1}^{K_n} \mathbb{E}\left[\delta(c_k^{(n)} = i,\ c_{\pi(k)}^{(n)} = j)\right],$$

$$\omega_{ij} = \sum_{n=1}^{N} \sum_{k=1}^{K_n} \mathbb{E}\left[\delta(w_k^{(n)} = i,\ c_k^{(n)} = j)\right].$$

On large datasets, N which is the number of sentences can be very large, and so, EM is inefficient because it requires that inference is performed on the entire dataset at each iteration. We therefore consider the online variant proposed by Cappé and Moulines (2009): instead of recomputing the pseudocounts on the whole dataset at each iteration t, those pseudocounts are updated using only a small subset $\mathcal{B}_t$ of the data, to get

$$\tau_{ij}^{(t)} = (1-\alpha_t)\tau_{ij}^{(t-1)} + \alpha_t \sum_{n \in \mathcal{B}_t} \sum_{k=1}^{K_n} \mathbb{E}\left[\delta(c_k^{(n)} = i,\ c_{\pi(k)}^{(n)} = j)\right],$$

and

$$\omega_{ij}^{(t)} = (1-\alpha_t)\omega_{ij}^{(t-1)} + \alpha_t \sum_{n \in \mathcal{B}_t} \sum_{k=1}^{K_n} \mathbb{E}\left[\delta(w_k^{(n)} = i,\ c_k^{(n)} = j)\right],$$

where the scalars α_t are defined by $\alpha_t = 1/(a+t)^\gamma$ with $0.5 < \gamma \leq 1$. In the experiments, we used $a = 4$. We chose γ in the set $\{0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$.

3.2 Approximate inference

Inference is performed on trees using the sum-product message passing algorithm, a.k.a. belief

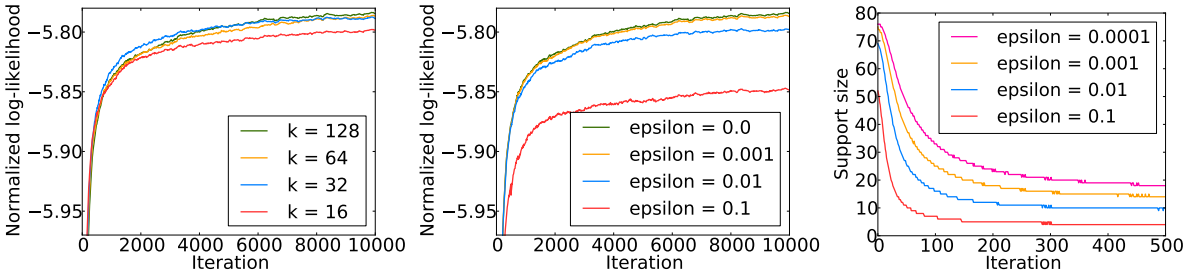

Figure 2: Comparison of the two projection methods for approximating vectors, for a model with 128 latent classes. The first two plots are the log-likelihood on a held-out set as a function of the iterates of online EM. Green curves ($k = 128$ and $\varepsilon = 0$) correspond to learning without approximation.

propagation, which extends the classical $\alpha - \beta$ recursions used for chains, see e.g. Wainwright and Jordan (2008). We denote by $\mathcal{N}(k)$ the set containing the children and the father of node k. In the exact message-passing algorithm, the message $\mu_{k \to \pi(k)}$ from node k to node $\pi(k)$ takes the form:

$$\mu_{k \to \pi(k)} = \mathbf{T}^\top \mathbf{u},$$

where $\mathbf{u}$ is the vector obtained by taking the elementwise product of all the messages received by node k except the one from node $\pi(k)$, i.e.,

$$u_i = \prod_{k' \in \mathcal{N}(k) \setminus \{\pi(k)\}} \mu_{k' \to k}(i).$$

Similarly, the pseudocounts can be written as

$$\mathbb{E}\left[\delta(c_k^{(n)} = i, c_{\pi(k)}^{(n)} = j)\right] \propto u_i T_{ij} v_j,$$

where $\mathbf{v}$ is the vector obtained by taking the elementwise product of all the messages received by node $\pi(k)$, except the one from node k, i.e.,

$$v_j = \prod_{k' \in \mathcal{N}(\pi(k)) \setminus \{k\}} \mu_{k' \to \pi(k)}(j).$$

Both these operations thus have quadratic complexity in the number of semantic classes. In order to reduce the complexity of those operations, we propose to start by projecting the vectors $\mathbf{u}$ and $\mathbf{v}$ on a set of sparse vectors, and then, perform the operations with the sparse approximate vectors. We consider two kinds of projections:

- *k-best projection*, where the approximate vector is obtained by keeping the k largest coefficients,
- *ε-best projection*, where the approximate vector is obtained by keeping the smallest set of larger coefficients such that their sum is greater than $(1 - \varepsilon)$ times the ℓ_1-norm of the original vector.

This method is similar to the one proposed by Pal et al. (2006). The advantage of the k-best projection is that we control the complexity of the operations, but not the error, while the advantage of the ε-best projection is that we control the error but not the complexity. As shown in Fig. 2, good choices for ε and k are respectively 0.01 and 16. We use these values in the experiments. We also note, on the right plot of Fig. 2, that during the first iterations of EM, the sparse vectors obtained with the ε-best projection have a large number of non-zero elements. Thus, this projection is not adequate to directly learn large latent class models. This issue is addressed in the next section, where we present a state splitting strategy in order to learn models with a large number of latent classes.

3.3 State splitting

A common strategy to speed up the learning of large latent state space models, such as ours, is to start with a small number of latent states, and split them during learning (Petrov, 2009). As far as we know, there are still no good heuristics to choose which states to split, or how to initialize the parameters corresponding to the new states. We thus apply the simple, yet effective method, consisting in splitting all states into two and in breaking the symmetry by adding a bit of randomness to the emission probabilities of the new states. As noted by Petrov (2009), state splitting could also improve the quality of learnt models.

3.4 Initialization

Because the negative log-likelihood function is not convex, initialization can greatly change the quality of the final model. Initialization for online EM is done by setting the initial pseudocounts, and then performing an M-step. We have considered

the following strategies to initialize our model:

- *random initialization:* the initial pseudocounts τ_{ij} and ω_{ij} are sampled from a uniform distribution on $[0, 1]$,
- *Brown initialization:* the model is initialized using the (normalized) pseudocounts obtained by the Brown clustering algorithm. Because a parameter equal to zero remains equal to zero when using the EM algorithm, we replace null pseudocounts by a small smoothing value, e.g., for observation i, we use $10^{-5} \times \max_j \omega_{ij}$,

4 Experiments

In this section, we present the datasets used for the experiments, and the two semi-supervised tasks on which we evaluate our models: named entity recognition and supersense tagging.

4.1 Datasets

We considered two datasets: the first one, which we refer to as the *music dataset*, corresponds to all the Wikipedia articles refering to a musical artist. They were extracted using the Freebase database[1]. This dataset comprises 2.22 millions sentences and 56 millions tokens. We choose this dataset because it corresponds to a restricted domain.

The second dataset are the articles of the NYT corpus (Sandhaus, 2008) corresponding to the period 1987-1997 and labeled as *news*. This dataset comprises 14.7 millions sentences and 310 millions tokens.

We parsed both datasets using the Stanford parser, and converted parse trees to dependency trees (De Marneffe et al., 2006). We decided to discard sentences longer than 50 tokens, for parsing time reasons, and then lemmatized tokens using Wordnet. Each word of our vocabulary is then a pair of lemma and its associated part-of-speech. This means that the noun *attack* and the verb *attack* are two different words. Finally, we introduced a special token, -*-, for infrequent (lemma, part-of-speech) pairs, in order to perform smoothing. For the music dataset, we kept the $25\ 000$ most frequent words, while for the NYT corpus, we kept the $100\ 000$ most frequent words. For the music dataset we set the number of latent states to 256, while we set it to 512 for the NYT corpus.

[1] www.freebase.com

4.2 Qualitative results

Before moving on to the quantitative evaluation of our model, we discuss qualitatively the induced semantic classes. Examples of semantic classes are presented in Tables 1, 2 and 3. Tree models with random initialization were used to obtain those semantic classes. First we observe that most classes can be easily given natural semantic interpretation. For example class 196 of Table 1 contains musical instruments, while class 116 contains musical genres.

Table 2 presents groups of classes that contain a given homonymous word; it seems that the different classes capture rather well the different senses of each word. For example, the word *head* belongs to the class 116, which contains body parts and to the class 127, which contains words referring to leaders.

4.3 Semi-supervised learning

We propose to evaluate and compare the different models in the following semi-supervised learning setting: we start by learning a model on the NYT corpus in an unsupervised way, and then use it to define features for a supervised classifier. We now introduce the tasks we considered.

4.3.1 Named entity recognition

The first supervised task on which we evaluate the different models, is named entity recognition. We cast it as a sequence tagging problem, and thus, we use a linear conditional random field (CRF) (Lafferty et al., 2001) as our supervised classifier. For each sentence, we apply the Viterbi algorithm in order to obtain the most probable sequence of semantic classes, and use this as features for the CRF. The only other feature we use is a binary feature indicating if the word is capitalized or not. Results of experiments performed on the MUC7 dataset are reported in table 4. The baseline for this task is assigning named entity classes to word sequences that occur in the training data.

4.3.2 Supersense tagging

Supersense tagging consists in identifying, for each word of a sentence, its corresponding supersense, a.k.a. lexicographer class, as defined by Wordnet (Ciaramita and Altun, 2006). Because each Wordnet synset belongs to one lexicographer class, supersense tagging can be seen as a coarse disambiguation task for nouns and verbs. We decided to evaluate our models on this task to

# 54	radio BBC television station tv stations channel 1 MTV program network fm music
# 52	chart billboard uk top top singles 100 Hot album country 40 10 R&B 200 US song u.s.
# 78	bach mozart liszt beethoven wagner chopin brahms stravinsky haydn debussy tchaikovsky
# 69	sound style instrument elements influence genre theme form lyric audience direction
#215	tour show concert performance appearance gig date tours event debut session set night party
#116	rock pop jazz classical folk punk metal roll hip country traditional -*- blues dance
#123	win receive sell gain earn award achieve garner give enjoy have get attract bring include
#238	reach peak hit chart go debut make top platinum fail enter gold become with certify
#203	piano concerto -*- for violin symphony in works sonata string of quartet orchestra no.
#196	guitar bass vocal drum keyboard piano saxophone percussion violin player trumpet organ
#243	leave join go move form return sign tour begin decide continue start attend meet disband
#149	school university college hall conservatory academy center church institute cathedral

Table 1: Selected semantic classes corresponding to the music dataset. Like LDA, our model is a probabilistic model which generates words from latent classes. Unlike LDA though, rather than treating words as exchangeable, it accounts for syntax and semantic relations between words. As a consequence, instead of grouping words with same topic but various semantic roles or grammatical functions, our model tends to group words that tend to be syntactically and semantically equivalent.

#116	**head** hand hands foot face shoulder way knee eyes back body finger car arms arm
#127	president member director chairman executive **head** editor professor manager secretary
#360	company corporation group industry fund **bank** association institute trust system
#480	street avenue side **bank** square precinct coast broadway district strip bridge station
#87	pay base sell use available buy depend make provide receive get lose spend **charge** offer
#316	**charge** arrest convict speak tell found accuse release die indict ask responsible suspend
#263	system computer machine technology plant product **program** equipment line network
#387	plan agreement contract effort **program** proposal deal offer bill bid order campaign request
#91	have be win score play lead hit make run -*- lose finish pitch start miss come go **shoot** take
#198	kill **shoot** die wound injure found arrest fire report take dead attack beat leave strike carry

Table 2: Semantic classes containing homonymous words. Different classes capture different senses of each word.

demonstrate the effect of homonymy. We cast supersense tagging as a classification problem and use posterior distribution of semantic classes as features for a support vector machine with the Hellinger kernel, defined by

$$K(\mathbf{p}, \mathbf{q}) = \sum_{c=1}^{C} \sqrt{p_c q_c},$$

where $\mathbf{p}$ and $\mathbf{q}$ are posterior distributions. We train and test the SVM classifier on the section A, B and C of the Brown corpus, tagged with Wordnet supersenses (SemCor). All the considered methods predict among the possible supersenses according to Wordnet, or among all the supersenses if the word does not appear in Wordnet. We report results in Table 5. The baseline predicts the most common supersense of the training set.

4.4 Discussion of results

First, we observe that hidden Markov models improve performances over Brown clustering, on both chains and trees. This seems to indicate that taking into account homonymy leads to richer models which is beneficial for both tasks. We also note that Brown clustering on dependency trees always outperforms Brown clustering on chains for the two tasks we consider, confirming that syntactic dependencies are a better structure to induce semantic classes than a linear chain.

Hidden Markov tree models also outperform hidden Markov chain models, except for supersense tagging on verbs. We believe that this drop in performance on verbs can be explained because in English the word order (Subject-Verb-Object) is strict, and thus, the chain model is able to dif-

#484	rise fell be close offer drop gain trade price jump slip end decline unchanged sell total lose
#352	it have would But be not nt will get may too make So see might can always still probably
#115	coach manager bill Joe george don pat Jim bob Lou al general mike Dan tom owner ray
#131	San St. santa Notre s Francisco calif. green tampa Diego louis class AP bay &aaa Fla. Jose
#350	strong short score good better hit second leave fast close impressive easy high quick enough
#274	A Another an new second single free -*- special fair national strong long major political big
#47	gogh rushdie pan guardia vega freud Prensa miserable picasso jesus Armani Monde Niro
#489	health public medical right care human civil community private social research housing
#238	building house home store apartment area space restaurant site neighborhood town park
#38	more very too as so much less enough But seem even because if particularly relatively pretty

Table 3: Randomly selected semantic classes corresponding to the news dataset.

	F1 score
Baseline	71.66
Brown clustering	82.57
tree Brown clustering	82.93
chain HMM, random init	84.66
chain HMM, Brown init	84.47
tree HMM, random init	84.07
tree HMM, Brown init	**85.49**

Table 4: Results of semi-supervised named entity recognition.

	nouns	verbs
Baseline	61.9 (0.2)	43.1 (0.2)
Brown clustering	73.9 (0.1)	63.7 (0.2)
tree Brown clustering	75.0 (0.2)	65.2 (0.2)
HMM (random)	76.1 (0.1)	63.0 (0.2)
HMM (Brown)	76.8 (0.1)	**66.6 (0.3)**
tree HMM (random)	76.7 (0.1)	61.5 (0.2)
tree HMM (Brown)	**77.9 (0.1)**	66.0 (0.2)

Table 5: Results of semi-supervised supersense tagging: prediction accuracies with confidence intervals, obtained on 50 random splits of the data.

ferentiate between subject and object, while the tree model treats subject and object in the same way (both are children of the verb). Moreover, in the tree model, verbs have a lot of children, such as adverbial clauses and auxiliary verbs, which share their transition probability distribution with the subject and the object. These two effects make the disambiguation of verbs more noisy for trees than for chains. Another possible explanation of this drop of performance is that it is due to errors made by the syntactic parser.

4.5 On optimization parameters

We briefly discuss the different choices that can influence learning efficiency in the proposed models. In practice, we have not observed noticeable differences between *ε-best projection* and *k-best projection* for the approximate inference, and we thus advise to use the latter as its complexity is controled. By contrast, as illustrated by results in tables 4 and 5, initialization can greatly change the performance in semi-supervised learning, in particular for tree models. We thus advise to initialize with Brown clusters. Finally, as noted by Liang and Klein (2009), the step size of online EM also has a significant impact on performance.

5 Conclusion

In this paper, we considered an arguably natural generative model of sentences for semantic class induction. It can be seen as a generalization of Brown clustering, taking into account homonymy and syntax, and thus allowed us to study their impact on semantic class induction. We developed an efficient algorithm to perform inference and learning, which makes it possible to learn in this model on large datasets, such as the New York Times corpus. We showed that this model induces relevant semantic classes and that it improves performance over Brown clustering on semi-supervised named entity recognition and supersense tagging. We plan to explore in future work better ways to model verbs, and in particular how to take into account the type of dependencies between words.

Acknowledgments

Francis Bach is supported in part by the European Research Council (SIERRA ERC-239993).

References

D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*.

J. L. Boyd-Graber and D. Blei. 2009. Syntactic topic models. In *Advances in Neural Information Processing Systems 21*.

P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*.

O. Cappé and E. Moulines. 2009. On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*.

M. Ciaramita and Y. Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the tenth conference of European chapter of the Association for Computational Linguistics*.

M. C. De Marneffe and C. D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*.

M. C. De Marneffe, B. MacCartney, and C. D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*.

P. S. Dhillon, D. Foster, and L. Ungar. 2011. Multi-view learning of word embeddings via CCA. *Advances in Neural Information Processing Systems*.

M. Faruqui, S. Padó, and M. Sprachverarbeitung. 2010. Training and evaluating a German named entity recognizer with semantic generalization. *Semantic Approaches in Natural Language Processing*.

D. Freitag. 2004. Trained named entity recognition using distributional clusters. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. 2005. Integrating topics and syntax. *Advances in Neural Information Processing Systems*.

G. Haffari, M. Razavi, and A. Sarkar. 2011. An ensemble model that combines syntactic and semantic clustering for discriminative dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*.

T. Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*.

R. Kneser and H. Ney. 1993. Improved clustering techniques for class-based statistical language modelling. In *Third European Conference on Speech Communication and Technology*.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the 18th International Conference on Machine Learning*.

W. Li and A. McCallum. 2005. Semi-supervised sequence modeling with syntactic topic models. In *Proceedings of the National Conference on Artificial Intelligence*.

P. Liang and D. Klein. 2009. Online EM for unsupervised models. In *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

P. Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

K. Lund and C. Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*.

S. Miller, J. Guinness, and A. Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT-NAACL*.

C. Pal, C. Sutton, and A. McCallum. 2006. Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *ICASSP 2006 Proceedings*.

S. Petrov. 2009. *Coarse-to-Fine Natural Language Processing*. Ph.D. thesis, University of California at Bekeley.

E. Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia*.

D. O. Séaghdha. 2010. Latent variable models of selectional preference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

O. Täckström, R. McDonald, and J. Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics*.

S. Tratz and E. Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

J. Uszkoreit and T. Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. *Proceedings of ACL-08: HLT*.

M. J. Wainwright and M. I. Jordan. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*.