Implementation of Spectral Learning of Latent-Variable Probabilistic Context-Free Grammars

Tatiana Matejovičová
Supervised by: Dr Mark-Jan Nederhof, Dr Louis Theran
University of St Andrews

Introduction

Probabilistic ontext-free grammars (PCFGs) are designed to model the syntax and statistical distribution of a particular language and are an integral part of multiple NLP applications. However, PCFG imposes probabilistic independence assumption on production rules which results in an insufficient representation of structural and syntactic dependencies of natural language. To improve this, latent states are assigned to nonterminal nodes during the training phase resulting in a greater parsing accuracy of the extracted grammar. Traditionally, Baum-Welsch algorithm is used to estimate hidden or latent variable models such as HMMs and CFGs. However, it gives no guarantee to find the global maximum of the likelihood function and therefore to give consistent parameter estimates. Spectral learning algorithm, based on sigular value decomposition, was proposed [1] as an alternative, resistant to local maxima of the likelihood function. In this project, the updated version of the original algorithm based on clustering [2] is implemented in Python. Furthermore, a pipeline to test the grammar accuracy is assembled and used in multiple experiments on training corpus size, number of latent states and training time.

L-PCFG Definition

An L-PCFG is an 8-tuple (N, I, P, m, n, t, q, π) where:

- N - a set of non-terminal symbols (non-terminals)
- $I \subset N$ - a set of in-terminals
- $P \subset N$ - a set of pre-terminals, $N = I \cup P$ and $I \cap P = \emptyset$
- $[n]$ - a set of terminal symbols, $[m]$ - a set of possible hidden states
- For all $a \in I$; $b,\ c \in N$ and $h_1,\ h_2,\ h_3,\ \in [m]$, there is a (binary) rule $a(h_1) \rightarrow b(h_2)\ c(h_3)$ with an associated probability
- For all $a \in P$, $h \in [m]$, $x \in [n]$, there is a (terminal) rule $a(h) \rightarrow x$ with an associated probability
- For all $a \in I$ and $h \in [m]$ there is probability of $a(h)$ being the root

Inputs

1. For each nonterminal node in each parse tree τ of the training corpus we obtain a training example of the form (t, o):
   - t - inside tree of τ (see 1b)
   - o - outside tree of τ (see 1c)

   For each nonterminal a define S^a the set of all corresponding training examples.
2. Feature functions of inside and outside trees:
   - $\phi : T \rightarrow \mathbb{R}^d$ - inside tree feature function maps an inside tree t to a vector $\phi(t) \in \mathbb{R}^d$
   - $\psi : O \rightarrow \mathbb{R}^{d'}$ - outside tree feature function maps an outside tree o to a vector $\psi(o) \in \mathbb{R}^{d'}$
3. An integer k denoting the SVD rank.
4. An integer m denoting the number of latent states.

Algorithm steps

1. **Singular Value Decomposition**
   - For all $a \in N$ calculate:

$$\Omega^a = \frac{1}{|S^a|} \sum_{(t,o) \in S^a} \phi(t)\psi(o)^T \in \mathbb{R}^{d \times d'}$$

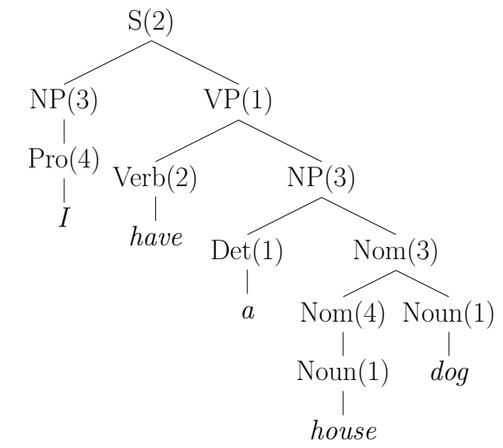
   - For all $a \in N$ calculate SVD of rank k on Ω^a.
   - Obtain:
     - $U^a \in \mathbb{R}^{k \times d}$ as a matrix of the left singular vectors of Ω^a corresponding to the k largest singular values
     - $V^a \in \mathbb{R}^{k \times d'}$ as a matrix of the right singular vectors of Ω^a corresponding to the k largest singular values
2. **Projection**
   For all $a \in N$:
   - For all training examples $(t, o) \in S^a$ compute:
     - $y = U^a\phi(t)$, $z = V^a\psi(o)$
     - Set x to be the concatenation of y and z
   - Define Z^a as the set of vectors x from training examples in S^a
3. **Clustering and Tree Annotation**
   - For all $a \in N$ cluster the set Z^a to m clusters using k-means clustering
   - Obtain a clustering function $\gamma : \mathbb{R}^{2k} \rightarrow [m]$ that maps a projected vector x to a cluster in $[m]$
   - Annotate each node in corpus with $\gamma(x)$ i.e. the latent state of the corresponding non-terminal node

Outputs

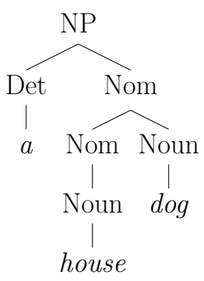
Compute the following to obtain the final parameters of the grammar:

- Probability of binary rules $p(a(h_1) \rightarrow b(h_2)\ c(h_3)|a(h_1))$ and terminal rules $p(a(h) \rightarrow x|a(h))$ as the relative frequency of their appearance in the annotated corpus.
- Probability of an annotated in-terminal symbol $a(h)$ being a root as the relative frequency of it being a root in the whole corpus.

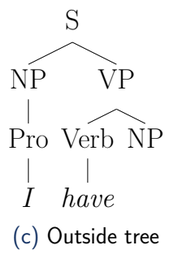
Example



(a) Parse tree annotated with latent states



(b) Inside tree



(c) Outside tree

Obr. 1

Intuition behind SVD

Singular value decomposition of Ω^a in the first step provides a way to approximate the matrix with one of lower rank. Thus, U^a and V^a can be used to project feature functions of inside and outside trees down to lower-dimensional vectors. Ω^a is an empirical estimate for the cross-covariance matrix between the inside and outside trees of a given nonterminal a [2]. Under the L-PCFG model, the inside and outside tree are independent given the latent state at their connecting point which means that latent state can be defined by finding patterns in inside and outside trees that occur together. Therefore by reducing the dimensions of this matrix by SVD, we get the representation that corresponds to the latent states.

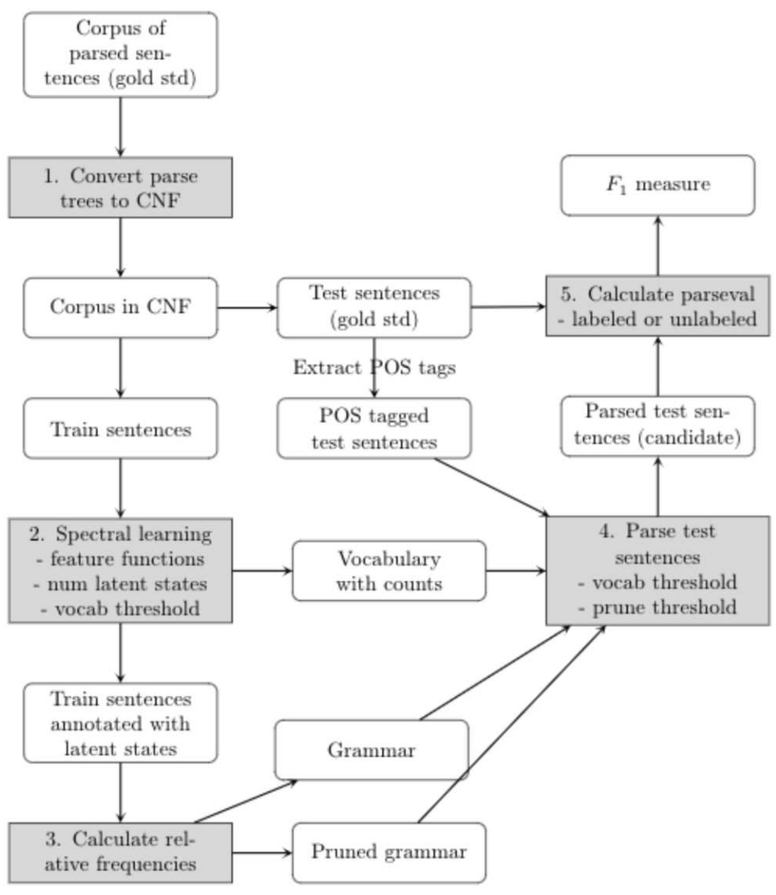
Implementation

Programming Language Python 3.6.4., Object-oriented design
Libraries

- Natural Language Toolkit (NLTK) provides an essential backbone for the software. Classes such as Tree and Nonterminal are heavily utilised.
- SciPy and its methods for sparse matrices are crucial for time feasibility when dealing with large and sparse matrices that were abundant in this project. NumPy is used for manipulation of dense vectors and matrices.
- Scikit-learn is a python machine learning library and its implementation of k-means clustering is utilised.

Corpus Wall Street Journal sections of Penn treebank

Pipeline

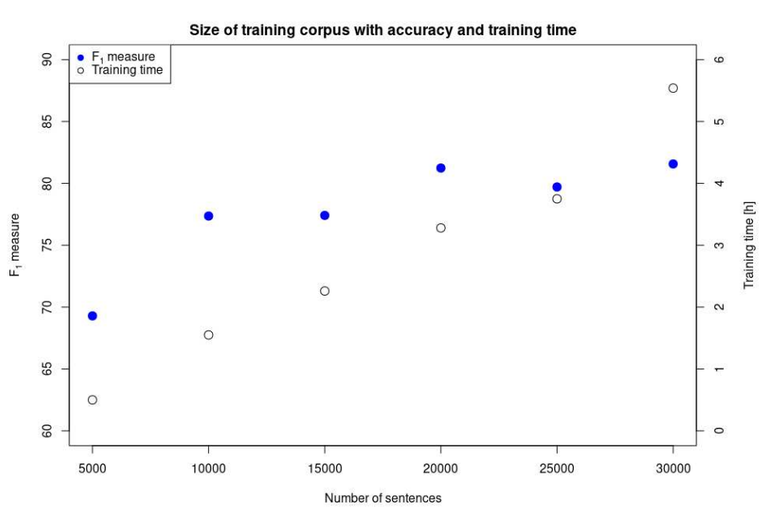


Obr. 2: Flowchart of the project pipeline

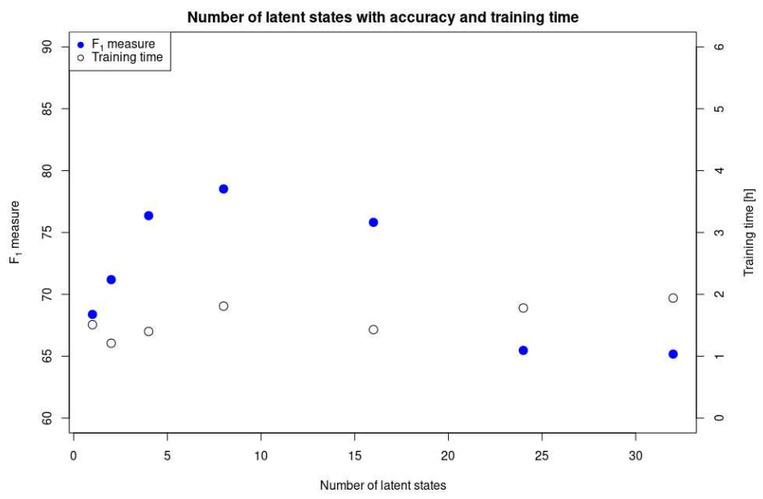
Experiment results

Experimental setup:

1. 8 latent states ($m = k = 8$), randomly selected 5 - 30 thousand sentences for training, 1 thousand for testing, experiment repeated 3 times and results averaged;
2. 1 - 32 latent states ($m = k$), randomly selected 10 thousand sentences for training, 1 thousand for testing, experiment repeated 3 times and results averaged;
3. Implementation compared to that of [3], 8 - 32 latent states ($m = k$), sections 1 - 21 of WSJ used for training, section 22 for testing;



Obr. 3: Experiment 1



Obr. 4: Experiment 2

Latent states	F_1 measure Spectral - previous	F_1 measure Spectral - this
8	85.6	78.46
16	87.77	77.87
24	88.53	76.15
32	88.82	72.73

Obr. 5: Experiment 3: Comparison to previous implementation

Conclusion

- Experiments have shown the expected properties in relation to training corpus size and number of latent states, proving the implementation correctness.
- With 8 latent states at least 20 thousand sentences are required to achieve close to optimal accuracy of about 80%.
- When trained on 10 thousand sentences 8 latent states were shown to result in optimal grammar.
- Grammars with 24 and 32 latent states performed worse than a PCFG model with no latent states which was likely the result of overfitting.
- Training time increases more rapidly with the size of the training corpus than the number of latent states.
- For 30 thousand sentences, the training completes in about 6 hours, which shows that the implementation is time feasible even with large corpora.
- The implementation performed worse than the original one for all numbers of latent states which was linked to multiple differences in the implementation and testing such as fewer feature functions and a distinct approach to POS tagging.

References

[1] Shay B Cohen, Karl Stratos, Michael Collins, Dean P Foster, and Lyle Ungar. Spectral learning of latent-variable pcfgs. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 223–231. Association for Computational Linguistics, 2012.

[2] Shashi Narayan and Shay B Cohen. Diversity in spectral learning for natural language parsing. *arXiv preprint arXiv:1506.00275*, 2015.

[3] Shay B Cohen, Karl Stratos, Michael Collins, Dean P Foster, and Lyle H Ungar. Experiments with spectral learning of latent-variable pcfgs. 2013.