Connectionist Language Model for Polish

Łukasz Brocki, Krzysztof Marasek, and Danijel Koržinek

Polish-Japanese Institute of Information Technology
Koszykowa 86, 02-008 Warszawa, Poland

Abstract. This article describes a connectionist language model, which may be used as an alternative to the well known n-gram models. A comparison experiment between n-gram and connectionist language models is performed on a Polish text corpus. Statistical language modeling is based on estimating a joint probability function of a sequence of words in a given language. This task is made problematic due to a phenomenon known commonly as the "curse of dimensionality". This occurs because the sequence of words used to test the model is most likely going to be different from anything present in the training data. Classic solutions to this problem are successfully achieved by using n-grams which generalize the data by concatenating short overlapping word sequences gathered from the training data. Connections models, however, can accomplish this by learning a distributed representation for words. They can simultaneously learn both the distributed representation for each word in the dictionary as well as the synaptic weights used for modeling the joint probability of word sequences. Generalization can be obtained thanks to the fact that if a sequence is made up of words that were already seen, it will receive a higher probability than an unseen sequence of words. In the experiments, perplexity is used as measure of language model quality.

Keywords: statistical language model, neural networks, multilayer perceptron.

1 Introduction

Statistical language modeling is especially important for automatic speech recognition and statistical machine translation where a language model is a crucial component for searching in the excessively large hypothesis space. The literature on the topic and the following paper describe a novel, connectionist language model which, in case of English, achieved better results than traditional, statistical n-gram models. The purpose of language modeling is to reduce the search space of the speech recognizer (decoder) by allowing only syntactically and semantically correct word sequences to be decoded. This is commonly achieved by two methods: formal grammars and statistical language models.

Grammars explicitly define which sequences are allowed. Algorithmically, they are represented by a Finite State Automaton, which is defined using a formal description often written in the Bachus-Naur Form (BNF) or some of its derivatives. Statistical language models offer greater flexibility than formal grammars, but at an increased uncertainty of the outcome.

R. Bembenik et al. (Eds.): Intelligent Tools for Building a Scient. Info. Plat., SCI 390, pp. 243–250.
springerlink.com

The task of the language model is to assign a probability to all the words in a dictionary depending on the previous words in a sequence. Instead of defining exactly which word sequences are allowed, they estimate the probability of a word sequence using a model trained on a large textual corpus. Usually only a short historical context is used, i.e. the model estimates only $P(w_t|w_{t-1})$ for bigram and $P(w_t|w_{t-1}, w_{t-2})$ for trigram language models. Given the sheer amount of words in a dictionary for any reasonable application of this model (at least several tens of thousands of words), even with such a small historical context, these models are usually very large. Because of this, the corpora used for training has to be unfeasibly large to make the model statistically significant and to deal with this problem many heuristics and tricks have to be used to make the model work well [9].

The goal of the work is to compare if the results with regards to connectionist language models shown in the literature apply equally well to the Polish language. Using a Polish language corpus, n-gram models are compared to the new connectionist approach described in sections 3 and 4. Section 5 contains the description and the results of the experiment comparing the two approaches. Section 6 contains the conclusions described by the achieved results and future research plans.

2 Dimensionality Curse

The language model must theoretically assign probabilities to all possible word combinations. The amount of these combinations increases exponentially with the amount of modeled words and length of the language model history. This problem is known as the curse of dimensionality. The amount of combinations is so large that the corpora, having even billions of words, contain only a small subset of all the possible word combinations in a given language. The language model trained on such data wouldn't work so well without extra treatment, because it would assign zero probability or an otherwise imprecise estimate to all the word sequences not present in the corpus.

The maximum likelihood method of training depends on the probabilities of word sequence (uni-gram, bi-gram, tri-gram, etc.) occurrences as measured from training data. Brown at al. [12] show that for the language model used in their experiments the maximum likelihood estimates will be 0 for 14.7 percent of the 3-grams and for 2.2 percent of the 2-grams in a new sample of English text. Generally, as n increases, the accuracy of an n-gram model increases, but the reliability parameter estimates, drawn from a limited training text, decreases.

The most often used solution to this problem is the reduction of accuracy of prediction by decreasing the context length. Jelinek and Mercer [13] propose interpolated estimation method that combines the estimates of several language models so as to use the estimates of the more accurate models where they are reliable and, where they are unreliable, to fall back on the more reliable estimates of less accurate models. Other often used methods are deleted interpolation

[2] and backing-off [3]. The use of these algorithms guarantees that each word sequence is going to have non-zero probability.

3 Connectionist Language Model

The connectionist models use a distributed representation of the items in the history and make much better use of contexts than interpolated or back-off n-gram models. They are much better in fighting the data sparseness problem, but also have the advantage that the model size grows only quasi-linearly when the context length is increased [14]. The general outline of the neural network performance when used for language modeling is following: the neural network calculates the contextual probabilities of words:

$$P(w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-n}) \tag{1}$$

using a distributed representation for each word in the dictionary. In distributed representation, each word is represented by a real number vector of certain length, generally several dozen dimensions. The information about the word sequences is distributed over several independent components. One could say that the neural network is a set of functions that transform the real valued vectors representing the previous word sequence, into probabilities of the occurrence of all the words in the dictionary. The probabilities of word occurrences in context calculated by the neural network depend on the feature vectors given to the input layer of the neural network. During the neural network training phase, both the network weights and values for the features of individual words in the dictionary are simultaneously set. The words that exist in the similar contexts will have similar features. The features are not a result of clustering analysis, but derived from automatic learning of dependencies between words using a gradient descent algorithm [1]. Such distributed representation of words allows for significant reduction in undesirable phenomena such as overtraining and overfit of the model to the data [16,17,18].

4 Formal Description of the Connectionist Language Model

The probability of the word sequence can be calculated when individual word frequencies in the context of previous words are known:

$$P(w_1, w_2, \ldots, w_{t-1}, w_t) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)\ldots \tag{2}$$

$$\ldots P(w_t|w_1, w_2, \ldots, w_{t-1}, w_t) \tag{3}$$

Feed forward neural networks use a context of size $n-1$, exactly as in the case of traditional n-grams:

$$P(w_t|w_{t-n+1}, \ldots, w_{t-1}) \tag{4}$$

Its worth noting that when n equals 1, the model is known as unigram and the probability of individual words is independent from context. A connectionist language model uses a dictionary where each word is assigned a feature vector C consisting of d real numbers representing the features of the given word. To calculate the probabilities of words next in the sequence, the neural network is fed a feature vector x consisting of a concatenation of the vectors:

$$C(w_t) \tag{5}$$

where w_t denotes a word number t in the sequence. The formula:

$$x = C(w_{t-n}), C(w_{t-n+1}), C(w_{t-n+2}), \ldots, C(w_{t-n+(n-1)}) \tag{6}$$

describes the connection of n final feature vectors in the word sequence - context of length n. The next step is to calculate the excitation of the feed forward neural network. The output layer consists of as many neurons as there are words in the dictionary plus one extra neuron which model the remaning words (i.e OOV or Out-Of-Vocabulary words). The output layer uses a softmax activation function [7]. The softmax function, described in formula 7, guarantees that the excitation of each neuron will lie in the range [0..1] and that the sum of all excitations will be equal to 1. This property allows interpreting the result of the neural network computation as the probabilities of individual words. The probability of the word represented by the neuron of index k equals:

$$P(w_t = k | w_{t-n}, w_{t-n+1}, \ldots, w_{t-n+(n-1)}) = \frac{e^{net\,k}}{\sum_{i=1}^{N} e^{net\,i}} \tag{7}$$

where net is the weighted sum of signals reaching the neuron, i.e. net value. Its worth noting that calculating the probability of any word requires the calculation of the weighted sum of signals for all the neurons of the network (the denominator of the equation above) and multiple use of a computationally expensive exponential function. In case of large vocabularies, such calculations may turn out to be considerably slower than their n-gram counterpart. The neural network is trained using a gradient descent algorithm in such a way as to minimize the log-likelihood of the word sequence:

$$L(\Theta) = \frac{1}{T} \sum_{t} \log f(w_t, w_{t-1}, \ldots, w_{t-n+1}; \Theta) \tag{8}$$

where Θ are the parameters of the language model, that is the weights of the neural network and word features. The gradient:

$$\frac{\delta L(\Theta)}{\delta \Theta} \tag{9}$$

can be calculated using the commonly known back propagation algorithm [6,7,10,15] for the training of a feed-forward multilayer perceptron (MLP) neural network. The algorithm needs to be modified, however, to calculate the gradient not only in the context of the synaptic weights, but also in the C matrix

containing the feature vectors representing the individual words. The model can be trained online, or using batch learning, e.g. by using 100 samples at a time. It's worth noting that unlike the synaptic weights, during an individual training iteration the matrix C will have most of its features' errors not calculated. Only after a long training and many iterations will all the words have their features modified.

5 Experiment

The fundamental problem of researchers endeavoring in the field of language modeling for the Polish language is the lack of substantial amounts of text corpora. The authors therefore decided to use the only corpus available, which is the IPI PAN corpus [11] in his experiments, which is a substantially large, morphosyntactically annotated corpus of general modern Polish language. The corpus was created by Zespół Inżynierii Lingwistycznej w Instytucie Podstaw Informatyki PAN as a part of projects of Komitet Badań Naukowych and statutory research of IPI PAN. The IPI PAN corpus is the first publicly accessible polish language corpus in the true meaning of the word corpus: it is a large, containing 250 million segments, collection of polish texts, morphosyntactically labeled, created according to modern standards and practices of large corpora development.

The experiments were performed using the transcripts of the fourth term session of the Polish parliament, which are a part of the IPI PAN corpus. Due to a long time needed to train and test the model, the transcripts that were randomly selected from the corpus of parliamentary meetings contained exactly 1005569 words. These were split into training (around 800 thousand words) and validation and test sets (around a 100 thousand each). The division of the data was also done randomly. In order to keep the dictionary size at an acceptable level and to have a reasonable statistical presence of all the words in the dictionary, the words that occurred in the corpus less than 9 times were not added to the dictionary (this value was established empirically). The final dictionary consisted of 10570 words. Two experiments were performed: the first used a feed forward neural network and the second was based on a traditional language model created using SRILM [5]. SRILM is de facto standard for n-gram modeling and contains a set of programs which support language model calculation and evaluation. It supports various smoothing and discounting techniques (e.g. Knesser-Ney [4,8], Witten-Bell, Good-Turing with individually adjustable parameters), large text corpora, class-based and cache models, etc. Model quality evaluation is perplexity based and can be computed counting all tokens in the test corpus or excluding end-of-sentence tags. Language model training using SRILM toolkit is very fast - for the experiment it took less than 1 minute on Pentium dual core processor. The first experiment was based on a feed forward neural network that had 250 inputs (5 words with 50 features each), 100 neurons in the hidden layer and 10508 outputs. It's worth nothing that the amount of outputs is larger by one than the size of the dictionary. That is because the model requires and extra neuron to

represent the words not in the dictionary - OOV. Each word in the dictionary had a unique vector of 50 real numbers representing the feature vector fed to the neural network. The initial values of the features were set to the range -0.01 to 0.01 using a random Gaussian distribution. The output layer used a softmax activation function. The model was trained using a gradient descent algorithm with parameter update momentum. The learning rate was set to 10^{-17} and the momentum to 0.99. These parameters were set empirically. Each time the parameters of the model were changed based on the calculated gradient, the change from the last iteration is multiplied by the momentum parameter and also added to the parameters. This is a known technique used to speed up the teaching process when the error gradient is stable in the subsequent iterations. Additionally, the usage of this technique increases the probability of escaping the local minima of the error function, as claimed by the researchers [7,10]. The training of the neural network based model took 16 days. The neural network achieved a perplexity of 164, which is around 20% better than the traditional n-gram model built using SRILM. Perplexity is a standard method for estimating the amount of fit the model has with the data in the test corpus. The lower the value, the better the model represents the underlying data. It has also been shown by many researchers that lower values of perplexity mean higher accuracy rates in speech recognition systems (e.g. [2]). Perplexity is calculated using the following formula:

$$Perplexity = 2^{-\sum_{i=1}^{N} \frac{1}{N} \log_2 p(x_i)} \tag{10}$$

The table below shows the results achieved in our experiments. It is worth noting that the context length was chosen as the optimum for their respective models: a context of 5 words for the MLP based model and 3 for the n-gram model. For n-gram models, larger contexts would require much more data than was available. Models based on MLPs require, however, a slightly larger context to function optimally. Similar results have been obtained by other researchers in the field [1]. N-gram model has been computed for several discounting and smoothing techniques giving similar results - perplexity on test data ranges from 193 to 194.

Table 1. Experimental results for the language model trained on the fourth term session of the Polish parliament

Model	Context	Perplexity
SRILM	3	193
MLP	5	164

6 Conclusions

This paper showed a connectionist language model trained on the task of Polish language. The results achieved by the authors for the Polish language are consistent with the results for the English language described in the literature. The

paper contains a general description of the connectionist language models. The experiments performed by the authors show that connectionist language models work better than traditional n-gram based models on the IPI PAN corpus.

Modeling of the context using a simple feed-forward network, also known as the multilayer perceptron, is less elegant than using recurrent neural networks. Feed forward networks have to observe the entire word context (i.e. several words at once) at each time interval. This approach is not as efficient and elegant because the increase of context also greatly increases the size of the network, which makes it loose the generalization potential and slows down its performance. In future works, the authors intend to use recurrent neural networks to eliminate this problem.

Acknowledgments. This work is a part of the SYNAT project financed by the Narodowe Centrum Badań i Rozwoju, contract no. SP/I/1/77065/10.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A Neural Probabilistic Language Model. In: NIPS 2000, vol. 13, pp. 933–938; revised in J. Machine Learning Research 3, 1137–1155 (2003)
2. Jelinek, F.: Statistical Methods for Speech Recognition. MIT Press, Cambridge (1998)
3. Katz, S.M.: Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. IEEE Transactions on Acoustics, Speech and Signal Processing 3, 400–401 (1987)
4. Kneser, R., Ney, H.: Improved backing-off for m-gram language modeling. In: International Conference on Acoustics, Speech and Signal Processing, pp. 181–184 (1995)
5. Stolcke, A.: SRILM - An Extensible Language Modeling Toolkit. In: Proc. Intl. Conf. Spoken Language Processing, Denver, Colorado (2002)
6. Duch, W., Korbowicz, J., Rutkowski, L., Tadeusiewicz, R.: Biocybernetyka i inżynieria biomedyczna, tom 6, Sieci neuronowe. Akademicka Oficyna Wydawnicza EXIT, Warszawa (2000)
7. Bishop, C.M.: Neural networks for pattern recognition. Oxford University Press (1995)
8. Ney, H., Kneser, R.: Improved clustering techniques for class-based statistical language modeling. In: European Conference on Speech Communication and Technology (Eurospeech), Berlin, pp. 973–976 (1993)
9. Chen, S.F., Goodman, J.T.: An empirical study of smoothing techniques for language modeling. Computer, Speech and Language 13(4), 359–393 (1999)
10. Tadeusiewicz, R.: Sieci neuronowe. Akademicka Oficyna Wydawnicza RM, Warszawa (1993)
11. Przepiórkowski, A.: Korpus IPI PAN. Wersja wstępna / The IPI PAN Corpus: Preliminary version. IPI PAN, Warszawa (2004)
12. Brown, P.F., DeSouza, P.V., Mercer, R.L., Della Pietra, V.J., Lai, J.C.: Class-based n-gram Models of Natural Language. Computational Linguistics 18, 467–479 (1992)
13. Jelinek, E., Mercer, R.L.: Interpolated estimation of Markov source parameters from sparse data. In: Proceedings, Workshop on Pattern Recognition in Practice, Amsterdam, The Netherlands, pp. 381–397 (1980)

14. Xu, P., Emami, A., Jelinek, F.: Training connectionist models for the structured language model. In: Proceedings of The 2003 Conference On Empirical Methods In Natural Language Processing (EMNLP 2003), pp. 160–167. Association for Computational Linguistics, Stroudsburg (2003)
15. Tan, C.N.W., Wittig, G.E.: A study of the parameters of a backpropagation stock price prediction model. In: Proceedings, First New Zealand International Artificial Neural Networks and Expert Systems (1993)
16. Niesler, T.R., Whittaker, E.W.D., Woodland, P.C.: Comparison of part-of-speech and automatically derived category-based language models for speech recognition. In: International Conference on Acoustics, Speech and Signal Processing, pp. 177–180 (1998)
17. Hinton, G.E.: Learning Distributed Representations of Concepts. In: Proceedings of the Eighth Annual Conference of the Cognitive Science Society, pp. 1–12 (1986)
18. Hinton, G.E.: Connectionist Learning Procedures. Artificial Intelligence Journal 40, 185–234 (1989)