Convolutional Neural Networks for Semantic Classification of Fluent Speech Phone Calls

Fredrik Kjellgren and Joakim Nordström

Telia Uppsala
Strandbodgatan 1
753 23 Uppsala, Sweden
{fredrik.kjellgren, joakim.nordstrom}@teliacompany.com

Abstract

This paper presents results from research on the application and suitability of open-source machine learning and deep learning libraries for semantic classification of natural language. More specifically, Convolutional Neural Networks were implemented in Google's TensorFlow and built and trained according to state-of-the-art methodology. The performance of the resulting models were compared against baseline results obtained by training and evaluating Naive Bayes classifiers and Support Vector Machines, as well as against models that Telia has built and deployed themselves. The results indicate that such deep neural networks can be successfully built and trained to outperform the competition on these specific classification tasks, which involve up to 101 distinct target classes. Based on these results, we hypothesize that the new models can successfully be deployed and improve call-routing systems, and furthermore be used as a component of a larger and more sophisticated system that can perform multiple inference tasks in parallel to find deeper understanding of speech.

1. Introduction

Routing telephone calls based on fluent speech customer responses is still a challenging task. Despite increased performance of automatic speech recognition (ASR) systems, the degrees of freedom of input introduced by fluent speech descriptions demand more sophisticated natural language understanding (NLU) systems compared to traditional statistical methods.

The objective of this master thesis was to research whether open-source libraries for machine learning (ML) and deep learning (DL) can be used to build NLU models that are suitable for use and deployment in said context. Furthermore, the intention was to asses how such models perform compared to Telia's own models.

2. Related Work

Training and evaluating NLU models for classifying text with ML algorithms is a topic that has been researched extensively over the years. In this project, the suitability of Convolutional Neural Networks (CNNs) for the desired goal was investigated. Naive Bayes (NB) classifiers and Support Vector Machines (SVMs) were also trained and evaluated in order to establish baseline results for comparison purposes.

NB classifiers are commonly applied to establish baseline results thanks to their simplicity, efficiency, and relatively good results despite the assumption of conditional independence between every pair of features (Rennie et al., 2003). They do however not make a good fit for more sophisticated systems in part due to the fact that the modelling assumption often violates the circumstances under which most real-world events occur. This especially holds true due to the nature of text in which there clearly exists dependencies between words.

SVMs have for a long time been regarded as the state-of-the-art in text classification. These classifiers are fast, robust, works well during both linear and non-linear separation thanks to the kernel trick, can effectively cope with sparse data, and have a strong ability to generalize knowledge in high-dimensional spaces (Joachims, 1998). However, it has been shown (Bengio and LeCun, 2007) that non-parametric algorithms such as SVMs are limited in their ability to scale intelligent behavior to high-dimensional problems.

2.1 Convolutional Neural Networks

CNNs are a special kind of feed-forward neural networks that have an architecture that is based on early 60s research (Hubel and Wiesel, 1962) that made new discoveries on the arrangement of neurons in the visual cortex of cats. When originally proposed, new state-of-the-art performance was presented for multiple different classification tasks. Compared to the traditional fully-connected network architecture, CNNs have three distinguishing characteristic features in local receptive fields, shared weights, and sub-sampling which make them more efficient in terms of computational complexity (LeCun et al., 1998).

Incidentally, CNNs were originally designed to operate on images and to take the spatial structure of data into account, making these suitable for computer vision. Yet, by using a feature representation scheme known as word embeddings (Mikolov et al., 2013) that embeds words into a high-dimensional space, the same kind of convolution operations can be made on sentences to capture the one-dimensional spatial aspect of word ordering. This has led to new state-of-the-art results on a variety of natural language processing (NLP) tasks such as sentiment analysis (Kim, 2014), semantic parsing (Blunsom et al., 2011), and named entity recognition (Collobert et al., 2011).

These results are promising, but they have however mostly been in the form of proof-of-concepts on tasks in which written text with relatively few target classes are distinguished between. In real, live systems, it is not uncom-

Dataset	Number of Utterances	Semantic Categories	Vocabulary Size
A	77 291	69	5 333 (6 903)
B	70 662	94	7 199 (9 298)
C	34 434	46	2 328 (3 446)
D	33 715	101	4 043 (5 475)

Table 1: Some of the main characteristics of the four different datasets used during this research. The numbers in parenthesis indicate the value prior to pre-processing.

mon that one needs to be able to scale this behavior to many more target classes. Furthermore, spoken utterances tend to be more chaotic and less structured compared to written utterances. As such, this work attempts to bring these previously presented concepts to a complex real world problem.

3. Method and Implementation

3.1 Datasets and Pre-processing

Four different datasets were used during this project, each of which contains utterances and semantic categories. The utterances consist of phrases of spoken Swedish that have been transcribed by an ASR engine. Each utterance has subsequently been manually paired with a semantic category that reflects the intention of the utterance. The data originates from four different companies and have been collected through recorded telephone calls in an initial phase of deployment of the currently used systems. The content of these datasets is hence the same content that has been used by Telia to train and evaluate their own models prior to deployment in a live setting. The inclusion of multiple different datasets is motivated by the importance of getting a sense of how the ML algorithms under study perform with respect to different dataset characteristics. Table 1 gives an overview of some of the features of the datasets.

Prior to the feature extraction process, the raw data of the utterance transcript files was pre-processed in order to sanitize the data and reduce its dimensionality. This pipeline includes steps such as lowercasing, removal of punctuations and stopwords, conversion of numbers into a designated number placeholder tag, and application of stemming.

3.2 Features

Features were extracted using a bag-of-words (BoW) approach with binary and term frequency-inverse document frequency weighting schemes for the baseline algorithms. The CNNs were however designed to take word embeddings as their input data in order to supply these with more dense feature vectors that overcome the loss of information on word order and semantic relatedness caused by BoW. In this project, two types of word embeddings were subject to investigation. The first approach relies on a *precursor embedding layer built into the network*. The weights of this layer were randomly initialized and learnt during training of the downstream classification task. The second approach instead *injects externally trained embeddings into this embedding layer*. This is accomplished by using an implementation of Google's word2vec (Mikolov et al., 2013) to learn embeddings on in-domain data in an unsupervised fashion.

Model	A	B	C	D
BNB	90.69	85.52	86.25	88.54
MNB	91.27	86.13	86.22	88.80
Lin-SVM	94.55	89.80	91.23	92.61
RBF-SVM	94.91	90.10	92.03	92.37
CNN-rand	**95.25**	90.52	**92.30**	92.60
CNN-pre	94.89	90.49	92.00	92.52
Telia	95.05	**91.07**	91.98	**93.11**

Table 2: Results in terms of classification accuracy. **BNB:** Bernoulli NB. **MNB:** Multinomial NB. **Lin-SVM:** Linear SVM. **RBF-SVM:** SVM with RBF kernel. **CNN-rand:** Convolutional Neural Network with randomly initialized embeddings. **CNN-pre:** Convolutional Neural Network with pre-trained embeddings. **Telia:** Telia's own model.

3.3 Hyperparameter Optimization and Evaluation

The relatively cheap computational costs involved when running the NB and SVM experiments paved way for extensive fine-tuning of model hyperparameters. This was approached by applying a technique called grid search, which enables exhaustive optimization that searches for an optimal model in a hyperparameter space defined by the Cartesian product of all possible hyperparameter values. This step as well as the model evaluation was bundled together to constitute a nested stratified 10-fold cross-validation in which the grid search is incorporated as well. Such an approach delivers results which are as unbiased as possible and that are minimally influenced by variance.

The CNNs are however much more expensive to optimize. Long training times combined with the fact that the number of possible combinations of hyperparameters grow exponentially and thus suffer from the curse of dimensionality make exhaustive parameter search infeasible. As such, the current approach has been to rely on a measure of heuristics and expert knowledge, as well as to apply moderately aggressive early-stopping methods, in order to make an educated guess on how to start searching for a good model configuration.

Models were evaluated according to a range of performance metrics which were measured on a macro scale. Accuracy, F1-score, precision, and recall are found among these. Furthermore, learning curve analysis is performed in order to analyze how each respective algorithm is affected by the amount of data that is available during training. This is an especially interesting aspect to research for the CNN, since deep learning algorithms are known to require a lot of data in order to produce good results.

4. Results

Some of the main results obtained during the experiments are shown in Table 2. Numbers in bold indicate the best observed score per dataset. These measurements were obtained using the same network architecture for all datasets. The selection was based on the fact that this architecture appeared to be relatively stable and provided good results across all datasets. This constraint is simply motivated by the computational costs involved in the hyperparameter tuning of CNNs and the limited duration of the project.

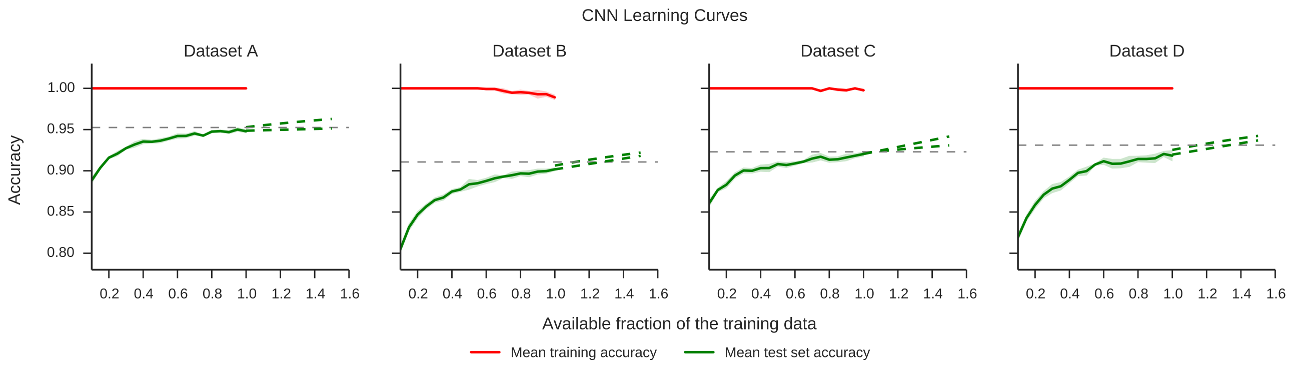

Figure 1: Results of the learning curve analysis. The plots illustrate the mean training and test set accuracies as functions of the available fraction of the training set.

Figure 1 depicts the results of the learning curve analysis. The green and dashed lines that continue past the 1.0 mark along the horizontal axis predict the future increase of accuracy as the dataset size is increased. These were produced by fitting an inverse power law function to the entire learning curve and fitting a linear function to the last four samples of it. The prediction of the polynomial fit was often overly optimistic, hence why the linear prediction was necessary in order balance it. The true progression is likely somewhere between the two lines.

5. Discussion

The results of Table 2 show that the CNNs and Telia's models tie in terms of best accuracy per dataset. The CNNs perform better on datasets A and C, whereas Telia's models does so on datasets B and D. Another interesting observation is that the pre-trained word embeddings gave no boost of performance. This is likely explained by the fact that the corpus on which these were trained was not large enough.

The learning curves of Figure 1 show that the CNNs would benefit from more data. More specifically, in order for these to reach and perform better than Telia's models on datasets B and D, an expansion of the dataset size by a factor of around 1.2 and 1.1 would be required. Furthermore, the learning curves show that the neural networks suffer from overfitting and would benefit from stronger regularization. This would however require additional hyperparameter tuning and was not feasible during the scope of the project. Despite this obvious overfitting, the CNNs perform well. This, along with the fact that the same architecture was used for all datasets, indicates that the presented results should be interpreted as soft lower margins of what this algorithm can achieve in this context.

6. Conclusions

The main goal of this master thesis was to research and assess the suitability of using open-source ML and DL libraries to build a specific type of NLU model. This model is aimed at routing fluent speech telephone calls by semantic classification of natural language. The results of the project show that the CNNs beat the baseline algorithms and put up with a good challenge with Telia's models. In fact, the CNNs beat Telia's models on two out of four datasets. The results should be interpreted as soft lower margins for what can be achieved due to the presence of overfitting and an insufficient amount of hyperparameter tuning due to time restrictions. Despite this, the proof of concept of the project is important and it was shown that CNNs are a very viable approach to accomplish the stated goal.

This also indicates that TensorFlow (Abadi et al., 2015), and likely also similar alternatives, can successfully be used to construct NLU models that are suitable for use in a call-routing context. We hypothesize that such a model may be suitable as a component of a larger and more sophisticated system that performs several different NLU tasks in parallel (e.g. with the addition of sentiment analysis, named entity recognition, etc.) in order to gain a deeper understanding of spoken utterances. In the context of customer support via telephone, this could potentially pave the way for construction of systems of higher intelligence that start to resemble that of a truly smart virtual assistant.

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdon, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, et al. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Y. Bengio and Y. LeCun. 2007. Scaling learning algorithms towards AI. In *Large-scale kernel machines*, 34.5.

J. Bergstra and Y. Bengio. 2012. Random Search for Hyper Parameter Optimization. In *Journal of Machine Learning Research*, pages 281–305.

P. Blunsom, N. de Freitas, E. Grefenstette, and K.M. Hermann. 2014. A deep architecture for semantic parsing. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural Language Processing (almost) from Scratch. In *Journal of Machine Learning Research*, vol. 12, pages 2493–2537.

D.H. Hubel and T.N. Wiesel. 1962. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. In *The Journal of physiology*, vol. 160, pages 106–154.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142.

Y. Kim. 2014. Convolutional Neural Networks for Sentence Classification. arXiv preprint arXiv:1408.5882.

Y. LeCun, B. Léon, Y. Bengio, and H. Patrick. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, vol. 86, pages 2278–2324.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.

J.D. Rennie, L. Shih, J. Teevan, and D.R. Karger. 2003. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. In *ICML*, vol. 3, pages 616–623.