CONVOLUTIONAL NEURAL NETWORKS CAN LEARN DURATION FOR DETECTING PITCH ACCENTS AND LEXICAL STRESS

Sabrina Stehwien, Antje Schweitzer, Ngoc Thang Vu

University of Stuttgart
sabrina.stehwien@ims.uni-stuttgart.de

Abstract: The duration of syllables or words is an important correlate of prosody and often used as a feature for automatic pitch accent detection. We have previously introduced a method for pitch accent detection using a convolutional neural network (CNN) that yields good performance using low-level acoustic descriptors alone, without any explicit duration information. In this paper, we use this model for various pitch accent and lexical stress detection tasks at the word and syllable level on the DIRNDL German radio news corpus. We show that information on word or syllable duration is encoded in the high-level CNN feature representation by training a linear regression model on these features to predict duration. The fact that this can be approximated suggests that the CNN makes use of implicit duration information that is derived from the frame-based input. We also observe that duration is only learnt in tasks where it is directly correlated with the target label. We compare two different methods of pooling that capture the input information differently and show how this affects what is encoded in the output representation.

1 Introduction

The duration of syllables or words is one of the primary correlates of prosody and thus often used as a feature in the automatic modelling of prominence. At the syllable level, stressed syllables are longer than unstressed ones, and pitch accents lengthen them additionally. At the word level, duration is useful since pitch accented words are not only more prominent due to lengthening, but also because content words such as verbs and nouns are longer on average and also tend to be accented more frequently [1].

Recently, neural networks have become a popular approach to detecting prosodic events and lexical stress automatically [2, 3, 4]. We have previously introduced a model for pitch accent detection using a convolutional neural network (CNN) that yields good performance using low-level acoustic descriptors alone [5]. We have reported results on various types of English speech data. The only preprocessing required for this method is time-alignment at the word level and the extraction of simple frame-based features. By default, it does not include any explicit duration information. Despite this simple setup, its performance is comparable to other methods.

In this paper, we report experimental results of various pitch accent and lexical stress detection tasks on German read speech. We tested two assumptions: First, since the duration of the input words or syllables is provided implicitly as the length of the frame sequence, we assume that it is likely that the CNN is making use of this information. Second, we expect that this depends on the method of pooling and padding. Analyzing what a CNN has learnt, however, is not straightforward: The use of such methods is motivated by the notion of letting the model learn the best high-level features from low-level input. The result is a feature representation that is not readily interpreted. We approached this problem by training a linear

regression model on this feature representation to predict word and syllable duration. If the duration can be approximated, then we can conclude that the CNN has encoded this information in the high-level features; it has been "learnt". This approach is similar to methods previously used in natural language processing [6] and for analyzing speaker embeddings [7]. To the best of our knowledge, this is the first study to apply this type of analysis to prosodic modelling.

2 Data

We used a subset of the DIRNDL German radio news corpus [8] that is prosodically annotated with GToBI pitch accents and phrase boundaries. The corpus contains recordings of several professional female and male speakers that amount to nearly 5 hours of speech. Table 1 lists the number of words, syllables and the relevant prosodic events in the dataset.

We used the word- and syllable-level annotations as separate datasets for different tasks. Pitch accent detection distinguishes between the *accent* and *none* classes on either words or syllables. At the syllable level, we also performed lexical stress detection (primary stress) and classification of the following pitch accent types: H* (high pitch accent), H*L (falling pitch accent) and L*H (rising pitch accent). For the type classification tasks, we left out all syllables belonging to the negative class since we were interested in the distinction of given accents.

words	35347	syllables	76396
accented	18137	stressed	35572
		accented	18958
		H*	16510
		H*L	6115
		L*H	7815

Table 1 – Overview of annotations used in the DIRNDL dataset.

3 Model

3.1 Convolutional neural network

Our pitch accent detector is a supervised learning method that labels each datapoint (word or syllable) with an event (pitch accent or stress). The model consists of a convolutional neural network (CNN) illustrated schematically in Figure 1. The network learns high-level feature representations from low-level acoustic input, resulting in a feature vector that is used for classification.

The input to the CNN is a matrix $X \in R^{d \times s}$ representing the current word ($w_{current}$) and the left and right context words ($w_{previous}$ and w_{next}). The d-dimensional feature vector for each frame consists of low-level descriptors of the audio signal. The network contains two convolution layers in which a set of 2-dimensional kernels are shifted across the input matrix. The output of each convolution layer is a set of feature maps corresponding to the number of kernels in the respective layer. Afterwards, max pooling selects the neuron with the highest activation in each feature map [9]. The output values are concatenated to one final output representation. For regularization, we also apply dropout [10] to this last layer. The feature vector is fed into the softmax activation layer which either has two units for binary classification or several units for multi-class classification.

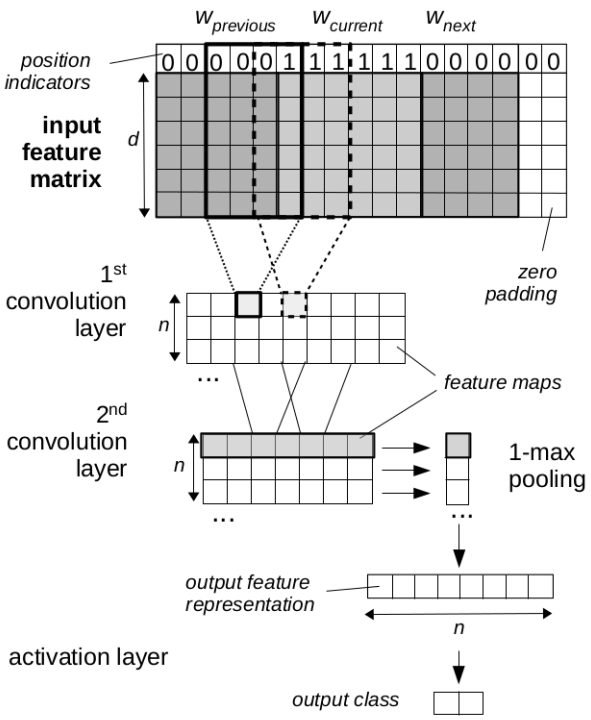

Figure 1 – CNN for binary classification (2 output units) using 1-max pooling. The input is a 3-word context window of acoustic and position indicator features.

3.2 Acoustic features

The speech signal was represented by the following low-level acoustic descriptors extracting using the OpenSMILE toolkit [11]: energy, loudness, zero-crossing rate (all 20 ms frames), F0, voicing probability and harmonics-to-noise-ratio (all 50ms frames). All features were extracted with a 10ms shift size. The features were z-scored per utterance file, which provides not only normalization per speaker but also per individual recording.

3.3 Hyperparameters and training

The first layer of the CNN consists of $n = 100$ 2-dimensional kernels of the shape $6 \times d$ and a stride of 4×1, with $d = 6$ as the number of acoustic features. The second layer consists of 100 kernels of the shape 4×1 and a stride of 2×1. We set the dropout rate to $p = 0.8$. The models are trained for 20 epochs with an adaptive learning rate (Adam [12]) and $l2$ regularization. We tracked the validation accuracy on the development set and applied the best model of the 20 epochs to the test data.

The data was divided into training, development and test splits with 10-fold cross validation. 500 held-out words were used as the development set and the test sets consisted of 1000 words. All experiments were repeated 3 times and the results were averaged. The CNN was implemented using the Keras[1] API.

4 Comparison of pooling methods

4.1 1-max pooling with position indicators

1-max pooling is a simple selection of the maximum value in the 2-dimensional pooling window which spans the length of each feature map (see Figure 1). The output of the pooling layer is

[1] `https://keras.io/`

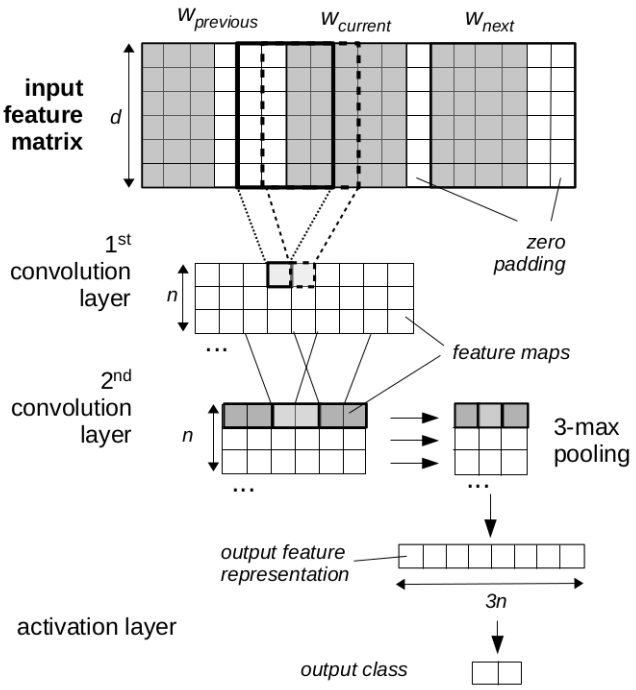

Figure 2 – CNN for binary classification and 3-max pooling. Each input word is padded separately.

a feature matrix of the shape $n \times 1$, which corresponds to the number of kernels in the last convolution layer. Our previous study [5] had shown that the pitch accent detector relies on a position indicator feature that distinguishes frames that pertain to the current word from those of the preceding and following context words. This can be explained as follows. Since the input words have different lengths, the input is padded with zeros so that each input matrix has the same size. The padding was concatenated to the end of the matrix. Therefore, there is no fixed position that corresponds to each individual input word. The position indicators help the model select features that are most important for representing the current word. They were added as an additional feature to the acoustic input and consist of ones for $w_{current}$ and zeros for the neighbouring words. In the first convolution layer, we set the kernel size to span the complete feature dimension $(d + 1)$ so that the position indicators are constantly taken into account.

4.2 3-max pooling

We compared the above method to an alternative that we refer to as 3-max pooling. In this setting, we zero padded each word in the input matrix separately so that each word has the same size (see Figure 2). Instead of pooling over each entire feature map, we defined three equal sized pooling windows for which one maximum value is selected. This method ensures that the learnt feature representation contains information on all three words. Therefore, the position indicator features should not be required.

The resulting flattened feature vector has the length $3n$. In these experiments, we used 30 feature maps, that is the output vector has a length of 90 and is thus comparable in size to the 1-max pooling output.

4.3 Experimental results

In sum, we obtained two different pooling methods and two different ways of padding the input. In the following experiments, we tested various combinations with and without the position indicator features.

Table 2 shows the F1-score for word-based pitch accent detection. As shown previously,

1-max pooling performs best using position indicators. The model does not perform well if they are left out, but the performance can be increased if the padding is placed in between the words. This may be due to the fact that the padded areas give an indication of word boundaries. 3-max pooling, as we had assumed, makes the position indicators unnecessary. Overall, 1-max pooling combined with the position indicators yields slightly better results, however the performance of both methods is comparable. One possible disadvantage of pooling over each word separately is that the zero padding breaks up the input signal. This still may not explain the differences observed in these experiments. Using $n = 100$ for 3-max pooling (that is, the final feature vectors has a length of 300), for example, increases the F1-score by roughly 1 percentage point.

setting		+ pos.-ind.	- pos.-ind.
1-max pooling	padding at end	**86.8**	63.4
	padding between words	86.2	70.4
3-max pooling	padding between words	85.8	**86.0**

Table 2 – Performance (F1-score) of word-based pitch accent detection using 1-max pooling ($n = 100$) and 3-max pooling ($n = 30$)

4.4 Task performance

Next we applied the two best settings (1-max pooling with position indicators and 3-max pooling without) on all word and syllable-level tasks. The results are listed in Table 3. The numbers for word-based pitch accent detection are the same as in Table 2 and included for comparison. Across all tasks, 1-max pooling performs slightly better than 3-max pooling, namely around 1-2 percentage points. The only exception is syllable-based pitch accent detection (2), where this difference is larger.

Syllable-level pitch accent detection is more difficult than word-based pitch accent detection for various reasons. First, the classes are less balanced (see Table 1), which makes it harder to model the minority class. Second, since the words are simply replaced by syllables in this case, much less context is provided (as discussed in [13]). Another reason, assuming that the model can learn duration, may be that the model cannot make use of any correlations to word length that could be attributed to word identity or part of speech. We also note that the CNN model had been optimized for word-level applications, while the aim of these experiments is a proof of concept, which does not require state-of-the-art performance.

Lexical stress detection (3) is an easier task, which is partly due to the fact that the classes are more balanced. Pitch accent detection is facilitated when only stressed syllables are considered (4). In this case, the model does not have to learn to distinguish unstressed and therefore unaccented syllables, which greatly reduces the number of negative examples in the data. For the classification of pitch accent types, we restrict the number of datapoints further and consider only syllables which carry a positive label. The three-way classification (5) is considerably more difficult than the binary distinction in (6), not only due to the multi-class learning problem but also since the H*-class makes up around half of the labels in (5) and thus constitutes a proportionally large majority class.

Task	1-max	3-max	*maj. class*
word-based			
(1) PA detection	86.8	86.0	*51.3*
syllable-based			
(2) PA detection	60.0	55.0	*24.8*
(3) stress detection	69.0	67.0	*46.6*
(4) PA detection stressed-only	75.0	73.0	*53.3*
(5) H*/H*L/L*H	66.0	65.0	*54.2*
(6) H*L/L*H	72.0	70.0	*56.1*

Table 3 – F1-scores and majority class sizes for various pitch accent (PA) and lexical stress detection tasks at the word and syllable level. The table compares 1-max pooling with position indicators to 3-max pooling without position indicators.

5 Evidence of duration in CNN output representations

5.1 Analysis using linear models

In order to investigate what the CNN has learnt, we analyzed the high-level output representation, which we refer to as the CNN features in this section. The features were extracted after the max pooling layer, before dropout, and consist of an n-dimensional vector for each data point: $n = 100$ for the 1-max setting, and $3n = 90$ for 3-max pooling. We used this data to train a linear regression model to predict the duration of each of the three input words ($w_{prev}, w_{cur}, w_{next}$) and measure the goodness of fit using the adjusted R^2. This can provide an idea of how well duration information is encoded in the CNN features.

Using this method, we compared models trained using 1-max pooling and position indicators and 3-max pooling without position indicators, assuming that the duration of the three input words (or syllables) was learnt differently. We expected to find differences across the various tasks listed in Table 3, where duration may be more or less important. For comparison, we also measured the Spearman correlation between the target label (the respective pitch accent or stress label) and the duration of w_{cur}. Both methods were implemented using R [14].

5.2 Results

The results of the analysis for each task are shown in Table 4. For word-based pitch accent detection (1), the linear model yields a moderately good fit (0.64). Compared to this, the R^2 for the duration of the current syllable is lower (<0.50 in tasks 2-4), but still high enough to be considered an approximate fit. Based on this result, we conclude that the CNN can learn duration, even if it is not directly included as a feature.

To show that this effect is due to the position indicators, we added results for 1-max pooling without them for comparison. The R^2 for w_{cur}, in this case, is much lower. This may lead to the assumption that the position indicators simply make the CNN "ignore" the context, however, our previous experiments [5] showed that this performed better than when not using any context at all.

Across tasks, the correlation (ρ) between the duration of w_{cur} and the target label appears to explain how well the former is predicted by the linear models. It also shows that duration is more indicative of pitch accents for words than it is for syllables, which is likely due to a lower variation in syllable length. Nevertheless, the results reflect the fact that there is a difference in length between accented and non-accented syllables, even when considering only stressed ones (3). Interestingly, duration appears to be less important for detecting lexical stress (2). There is no correlation between the syllable duration and target label for the tasks that classify

given pitch accent types (5-6), and thus the CNN features do not appear to encode much of this information.

Both pooling methods lead to a similar R^2 on w_{cur} as well as similar performance levels (shown in Table 3). The only notable difference is the fit on w_{prev} and w_{next}. When using 1-max pooling, the CNN can not be said to learn the duration of the two context words, since R^2 is very low. For 3-max pooling, however, these numbers are increased. In this case, the CNN features show evidence of containing duration information for all three input words, but similar to 1-max pooling, the duration of w_{cur} yields the best fit. This is an interesting result, since there were no position indicators used in this setting. Thus, the CNN appears to have automatically learnt features pertaining mainly to the "correct" word or syllable.

task/setting		1-max pooling	3-max pooling
measure	ρ	R^2	R^2
duration	w_{cur}	w_{prev} / w_{cur} / w_{next}	w_{prev} / w_{cur} / w_{next}
word-based			
(1) PA detection	0.70	0.11 / 0.64 / 0.09	0.48 / 0.61 / 0.41
- pos.-ind.		0.06 / 0.14 / 0.06	
syllable-based			
(2) PA detection	0.34	0.06 / 0.42 / 0.06	0.16 / 0.40 / 0.18
(3) stress detection	0.22	0.11 / 0.31 / 0.07	0.31 / 0.38 / 0.32
(4) PA detection str.-only	0.36	0.09 / 0.40 / 0.06	0.24 / 0.38 / 0.21
(5) H*/H*L/L*H	-0.04	0.05 / 0.16 / 0.12	0.21 / 0.19 / 0.15
(6) H*L/L*H	-0.04	0.03 / 0.14 / 0.07	0.09 / 0.17 / 0.08

Table 4 – R^2 of predicting word duration using the output of CNN models trained on various tasks. The table compares 1-max pooling with position indicators ($n = 100$) and 3-max pooling without position indicators ($n = 30$). ρ refers to the Spearman correlation between the duration of w_{cur} and the target label (pitch accent or stress).

6 Conclusion

In this paper, we described experiments and results indicating that a CNN-based pitch accent detector can learn duration information on its own, even though the input consists of simple frame-based features without explicit information on word length. This also holds for syllable-based pitch accent and lexical stress detection as long as it helps to solve the respective task. We compared two pooling methods and showed that pooling over each of the three input words or syllables, as opposed to simple 1-max pooling, increases the amount of context information captured by the CNN. Most information that is learnt, however, pertains to the current word or syllable. In this study, we focused only on the learning of duration. We are currently analyzing what information derived from the frame-based acoustic features is encoded in the CNN output.

So far we have only provided evidence that the CNN output contains specific information, that is the duration of syllables and words. What these experiments cannot answer is the question of how exactly this information is encoded in the final feature representation and how it is learnt. This is a current challenge for research on neural networks and is currently attracting considerable attention. In our case, this work constitutes a first step that contributes to our understanding of neural-network-based models of prosody.

References

[1] BATLINER, A., E. NÖTH, J. BUCKOW, R. HUBER, V. WARNKE, and H. NIEMANN: *Duration features in prosodic classification: Why normalization comes second, and what they really encode*. In *Proceedings of the ISCA Tutorial and Research Workshop on Speech Recognition and Understanding*, pp. 23–28. 2001.

[2] ROSENBERG, A., R. FERNANDEZ, and B. RAMABHADRAN: *Modeling phrasing and prominence using deep recurrent learning*. In *Proceedings of Interspeech*, pp. 3066–3070. 2015.

[3] SHAHIN, M., J. EPPS, and B. AHMED: *Automatic classification of lexical stress in English and Arabic languages using deep learning*. In *Proceedings of Interspeech*, pp. 175–179. 2016.

[4] LI, K., S. MAO, X. LI, Z. WU, and H. MENG: *Automatic lexical stress and pitch accent detection for l2 english speech using multi-distribution deep neural networks. Speech Communication*, 96, pp. 28–36, 2018.

[5] STEHWIEN, S. and N. T. VU: *Prosodic event recognition using convolutional neural networks with context information*. In *Proceedings of Interspeech*. 2017.

[6] ADI, Y., E. KERMANY, Y. BELINKOV, and Y. GOLDBERG: *Analysis of sentence embedding models using prediction tasks in natural language processing. IBM Journal of Research and Development*, 61(4/5), pp. 3–1, 2017.

[7] WANG, S., Y. QIAN, and K. YU: *What does the speaker embedding encode?* In *Proceedings of Interspeech*, pp. 1497–1501. 2017.

[8] ECKART, K., A. RIESTER, and K. SCHWEITZER: *A discourse information radio news database for linguistic analysis*. In S. H. CHRISTIAN CHIARCOS, SEBASTIAN NORDHOFF (ed.), *Linked Data in Linguistics. Representing Data and Language Metadata*, pp. 65–75. Springer Heidelberg, 2012.

[9] CIRESAN, D. C., U. MEIER, J. MASCI, L. M. GAMBARDELLA, and J. SCHMIDHUBER: *Flexible, high-performance convolutional neural networks for image classification*. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 2011.

[10] SRIVASTAVA, N., G. HINTON, A. KRIZHEVSKY, I. SUTSKEVER, and R. SALAKHUTDINOV: *Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research*, 15(1), pp. 1929–1958, 2014.

[11] EYBEN, F., F. WENINGER, F. GROSS, and B. SCHULLER: *Opensmile – the Munich versatile and fast open-source audio feature extractor*. In *Proceedings of the ACM Multimedia*, pp. 1459–1462. 2010.

[12] KINGMA, D. P. and J. BA: *Adam: A method for stochastic optimization*. arXiv preprint arXiv:1412.6980, 2017.

[13] ROSENBERG, A. and J. HIRSCHBERG: *Detecting pitch accents at the word, syllable and vowel level*. In *HLT-NAACL*. 2009.

[14] R CORE TEAM: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2013. URL `http://www.R-project.org/`.