Sequential Short-Text Classification with Recurrent and Convolutional Neural Networks

Ji Young Lee*
MIT
jjylee@mit.edu

Franck Dernoncourt*
MIT
francky@mit.edu

Abstract

Recent approaches based on artificial neural networks (ANNs) have shown promising results for short-text classification. However, many short texts occur in sequences (e.g., sentences in a document or utterances in a dialog), and most existing ANN-based systems do not leverage the preceding short texts when classifying a subsequent one. In this work, we present a model based on recurrent neural networks and convolutional neural networks that incorporates the preceding short texts. Our model achieves state-of-the-art results on three different datasets for dialog act prediction.

1 Introduction

Short-text classification is an important task in many areas of natural language processing, including sentiment analysis, question answering, or dialog management. Many different approaches have been developed for short-text classification, such as using Support Vector Machines (SVMs) with rule-based features (Silva et al., 2011), combining SVMs with naive Bayes (Wang and Manning, 2012), and building dependency trees with Conditional Random Fields (Nakagawa et al., 2010). Several recent studies using ANNs have shown promising results, including convolutional neural networks (CNNs) (Kim, 2014; Blunsom et al., 2014; Kalchbrenner et al., 2014) and recursive neural networks (Socher et al., 2012).

Most ANN systems classify short texts in isolation, i.e., without considering preceding short texts. However, short texts usually appear in sequence (e.g., sentences in a document or utterances in a dialog), therefore using information from preceding short texts may improve the classification accuracy. Previous works on sequential short-text classification are mostly based on non-ANN approaches, such as Hidden Markov Models (HMMs) (Reithinger and Klesen, 1997), (Stolcke et al., 2000), maximum entropy (Ang et al., 2005), and naive Bayes (Lendvai and Geertzen, 2007).

Inspired by the performance of ANN-based systems for non-sequential short-text classification, we introduce a model based on recurrent neural networks (RNNs) and CNNs for sequential short-text classification, and evaluate it on the dialog act classification task. A dialog act characterizes an utterance in a dialog based on a combination of pragmatic, semantic, and syntactic criteria. Its accurate detection is useful for a range of applications, from speech recognition to automatic summarization (Stolcke et al., 2000). Our model achieves state-of-the-art results on three different datasets.

2 Model

Our model comprises two parts. The first part generates a vector representation for each short text using either the RNN or CNN architecture, as discussed in Section 2.1 and Figure 1. The second part classifies the current short text based on the vector representations of the current as well as a few preceding short texts, as presented in Section 2.2 and Figure 2.

We denote scalars with italic lowercases (e.g., k, b_f), vectors with bold lowercases (e.g., $\mathbf{s}$, $\mathbf{x}_i$), and matrices with italic uppercases (e.g., W_f). We

* These authors contributed equally to this work.

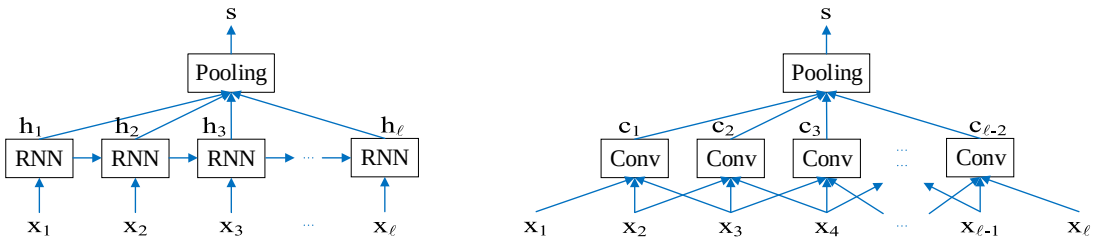

Figure 1: RNN (left) and CNN (right) architectures for generating the vector representation $\mathbf{s}$ of a short text $\mathbf{x}_{1:\ell}$. For CNN, Conv refers to convolution operations, and the filter height $h = 3$ is used in this figure.

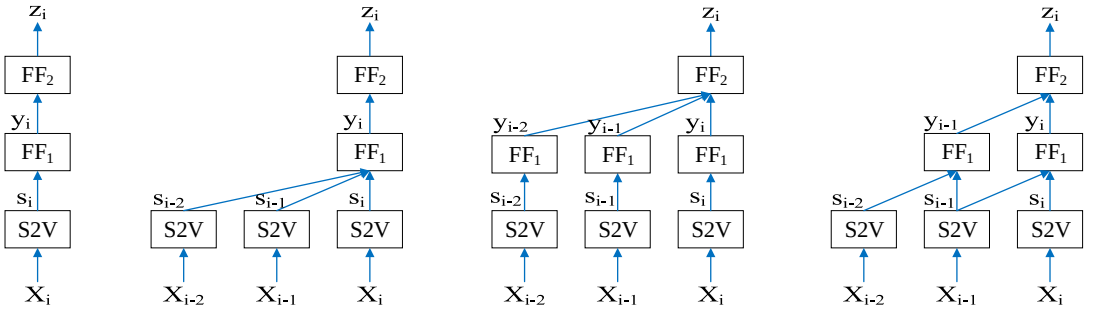

Figure 2: Four instances of the two-layer feedforward ANN used for predicting the probability distribution over the classes $\mathbf{z}_i$ for the i^{th} short-text $\mathbf{X}_i$. S2V stands for short text to vector, which is the RNN/CNN architecture that generates $\mathbf{s}_i$ from $\mathbf{X}_i$. From left to right, the history sizes (d_1, d_2) are $(0,0)$, $(2,0)$, $(0,2)$ and $(1,1)$. $(0,0)$ corresponds to the non-sequential classification case.

use the colon notation $\mathbf{v}_{i:j}$ to denote the sequence of vectors $(\mathbf{v}_i, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_j)$.

2.1 Short-text representation

A given short text of length ℓ is represented as the sequence of m-dimensional word vectors $\mathbf{x}_{1:\ell}$, which is used by the RNN or CNN model to produce the n-dimensional *short-text representation* $\mathbf{s}$.

2.1.1 RNN-based short-text representation

We use a variant of RNN called Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997). For the t^{th} word in the short-text, an LSTM takes as input $\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}$ and produces $\mathbf{h}_t, \mathbf{c}_t$ based on the following formulas:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(W_i\mathbf{x}_t + U_i\mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(W_f\mathbf{x}_t + U_f\mathbf{h}_{t-1} + \mathbf{b}_f) \\
\tilde{\mathbf{c}}_t &= \tanh(W_c\mathbf{x}_t + U_c\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \\
\mathbf{o}_t &= \sigma(W_o\mathbf{x}_t + U_o\mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}$$

where $W_j \in \mathbb{R}^{n\times m}$, $U_j \in \mathbb{R}^{n\times n}$ are weight matrices and $\mathbf{b}_j \in \mathbb{R}^n$ are bias vectors, for $j \in \{i, f, c, o\}$. The symbols $\sigma(\cdot)$ and $\tanh(\cdot)$ refer to the element-wise sigmoid and hyperbolic tangent functions, and $\odot$ is the element-wise multiplication. $\mathbf{h}_0 = \mathbf{c}_0 = \mathbf{0}$.

In the pooling layer, the sequence of vectors $\mathbf{h}_{1:\ell}$ output from the RNN layer are combined into a single vector $\mathbf{s} \in \mathbb{R}^n$ that represents the short-text, using one of the following mechanisms: last, mean, and max pooling. Last pooling takes the last vector, i.e., $\mathbf{s} = \mathbf{h}_\ell$, mean pooling averages all vectors, i.e., $\mathbf{s} = \frac{1}{\ell}\sum_{t=1}^{\ell}\mathbf{h}_t$, and max pooling takes the element-wise maximum of $\mathbf{h}_{1:\ell}$.

2.1.2 CNN-based short-text representation

Using a *filter* $W_f \in \mathbb{R}^{h\times m}$ of height h, a convolution operation on h consecutive word vectors starting from t^{th} word outputs the scalar *feature*

$$c_t = \text{ReLU}(W_f \bullet X_{t:t+h-1} + b_f)$$

where $X_{t:t+h-1} \in \mathbb{R}^{h\times m}$ is the matrix whose i^{th} row is $\mathbf{x}_i \in \mathbb{R}^m$, and $b_f \in \mathbb{R}$ is a bias. The symbol $\bullet$ refers to the dot product and $\text{ReLU}(\cdot)$ is the element-wise rectified linear unit function.

We perform convolution operations with n different filters, and denote the resulting features as $\mathbf{c}_t \in \mathbb{R}^n$, each of whose dimensions comes from a distinct filter. Repeating the convolution operations

for each window of h consecutive words in the short-text, we obtain $\mathbf{c}_{1:\ell-h+1}$. The short-text representation $\mathbf{s} \in \mathbb{R}^n$ is computed in the max pooling layer, as the element-wise maximum of $\mathbf{c}_{1:\ell-h+1}$.

2.2 Sequential short-text classification

Let $\mathbf{s}_i$ be the n-dimensional short-text representation given by the RNN or CNN architecture for the i^{th} short text in the sequence. The sequence $\mathbf{s}_{i-d_1-d_2:i}$ is fed into a two-layer feedforward ANN that predicts the class for the i^{th} short text. The hyperparameters d_1, d_2 are the history sizes used in the first and second layers, respectively.

The first layer takes as input $\mathbf{s}_{i-d_1-d_2:i}$ and outputs the sequence $\mathbf{y}_{i-d_2:i}$ defined as

$$\mathbf{y}_j = \tanh\left(\sum_{d=0}^{d_1} W_{-d}\,\mathbf{s}_{j-d} + \mathbf{b}_1\right),\ \forall j \in [i-d_2, i]$$

where $W_0, W_{-1}, W_{-2} \in \mathbb{R}^{k\times n}$ are the weight matrices, $\mathbf{b}_1 \in \mathbb{R}^k$ is the bias vector, $\mathbf{y}_j \in \mathbb{R}^k$ is the *class representation*, and k is the number of classes for the classification task.

Similarly, the second layer takes as input the sequence of class representations $\mathbf{y}_{i-d_2:i}$ and outputs $\mathbf{z}_i \in \mathbb{R}^k$:

$$\mathbf{z}_i = \text{softmax}\left(\sum_{j=0}^{d_2} W_{-j}\,\mathbf{y}_{i-j} + \mathbf{b}_2\right)$$

where $U_0, U_{-1}, U_{-2} \in \mathbb{R}^{k\times k}$ and $\mathbf{b}_2 \in \mathbb{R}^k$ are the weight matrices and bias vector.

The final output $\mathbf{z}_i$ represents the probability distribution over the set of k classes for the i^{th} short-text: the j^{th} element of $\mathbf{z}_i$ corresponds to the probability that the i^{th} short-text belongs to the j^{th} class.

3 Datasets and Experimental Setup

3.1 Datasets

We evaluate our model on the dialog act classification task using the following datasets:

- DSTC 4: Dialog State Tracking Challenge 4 (Kim et al., 2015; Kim et al., 2016).
- MRDA: ICSI Meeting Recorder Dialog Act Corpus (Janin et al., 2003; Shriberg et al., 2004). The 5 classes are introduced in (Ang et al., 2005).
- SwDA: Switchboard Dialog Act Corpus (Jurafsky et al., 1997).

For MRDA, we use the train/validation/test splits provided with the datasets. For DSTC 4 and SwDA, only the train/test splits are provided.[1] Table 1 presents statistics on the datasets.

Dataset	$\lvert C\rvert$	$\lvert V\rvert$	Train	Validation	Test
DSTC 4	89	6k	24 (21k)	5 (5k)	6 (6k)
MRDA	5	12k	51 (78k)	11 (16k)	11 (15k)
SwDA	43	20k	1003 (193k)	112 (23k)	19 (5k)

Table 1: Dataset overview. $|C|$ is the number of classes, $|V|$ the vocabulary size. For the train, validation and test sets, we indicate the number of dialogs (i.e., sequences) followed by the number of utterances (i.e., short texts) in parenthesis.

3.2 Training

The model is trained to minimize the negative log-likelihood of predicting the correct dialog acts of the utterances in the train set, using stochastic gradient descent with the Adadelta update rule (Zeiler, 2012). At each gradient descent step, weight matrices, bias vectors, and word vectors are updated. For regularization, dropout is applied after the pooling layer, and early stopping is used on the validation set with a patience of 10 epochs.

4 Results and Discussion

To find effective hyperparameters, we varied one hyperparameter at a time while keeping the other ones fixed. Table 2 presents our hyperparameter choices.

Hyperparameter	**Choice**	**Experiment Range**
LSTM output dim. (n)	100	50 – 1000
LSTM pooling	max	max, mean, last
LSTM direction	unidir.	unidir., bidir.
CNN num. of filters (n)	500	50 – 1000
CNN filter height (h)	3	1 – 10
Dropout rate	0.5	0 – 1
Word vector dim. (m)	200, 300	25 – 300

Table 2: Experiments ranges and choices of hyperparameters. Unidir refers to the regular RNNs presented in Section 2.1.1, and bidir refers to bidirectional RNNs introduced in (Schuster and Paliwal, 1997).

We initialized the word vectors with the 300-dimensional word vectors pretrained with word2vec on Google News (Mikolov et al., 2013a; Mikolov et al., 2013b) for DSTC 4, and the 200-dimensional

[1] All train/validation/test splits can be found at `https://github.com/Franck-Dernoncourt/naacl2016`

	d_2	LSTM			CNN		
d_1		0	1	2	0	1	2
DSTC4	0	63.1 (62.4, 63.6)	65.7 (65.6, 65.7)	64.7 (63.9, 65.3)	64.1 (63.5, 65.2)	65.4 (64.7, 66.6)	65.1 (63.2, 65.9)
	1	**65.8** (65.5, 66.1)	65.7 (65.3, 66.1)	64.8 (64.6, 65.1)	65.3 (64.1, 65.9)	65.1 (62.1, 66.2)	64.9 (64.4, 65.6)
	2	65.7 (65.0, 66.2)	65.5 (64.4, 66.1)	64.9 (64.6, 65.2)	65.7 (64.9, 66.3)	**65.8** (65.2, 66.1)	65.4 (64.5, 66.0)
MRDA	0	82.8 (82.4, 83.1)	83.2 (82.9, 83.4)	82.9 (82.4, 83.4)	83.2 (83.0, 83.4)	83.5 (82.9, 84.0)	83.8 (83.4, 84.2)
	1	83.2 (82.6, 83.7)	83.8 (83.5, 84.4)	83.6 (83.2, 83.8)	**84.6** (84.5, 84.9)	**84.6** (84.4, 84.8)	84.1 (83.8, 84.4)
	2	**84.1** (83.5, 84.4)	83.9 (83.4, 84.7)	83.3 (82.6, 84.2)	84.4 (84.1, 84.8)	**84.6** (84.5, 84.7)	84.4 (84.2, 84.7)
SwDA	0	66.3 (65.1, 68.0)	67.9 (66.3, 68.6)	67.8 (66.7, 69.0)	67.0 (65.3, 68.7)	69.1 (68.5, 70.0)	69.7 (69.2, 70.9)
	1	68.4 (67.8, 68.8)	67.8 (65.5, 68.9)	67.3 (65.5, 69.5)	69.9 (69.1, 70.9)	69.8 (69.3, 70.6)	69.9 (68.8, 70.6)
	2	**69.5** (68.9, 70.2)	67.9 (66.5, 69.4)	67.7 (66.9, 68.9)	**71.4** (70.4, 73.1)	71.1 (70.2, 72.1)	70.9 (69.7, 71.7)

Table 3: Accuracy (%) on different architectures and history sizes d_1, d_2. For each setting, we report average (minimum, maximum) computed on 5 runs. Sequential classification ($d_1 + d_2 > 0$) outperforms non-sequential classification ($d_1 = d_2 = 0$). Overall, the CNN model outperformed the LSTM model for all datasets, albeit by a small margin except for SwDA. We also tried a variant of the LSTM model, gated recurrent units (Cho et al., 2014), but the results were generally lower than LSTM.

word vectors pretrained with GloVe on Twitter (Pennington et al., 2014) for MRDA and SwDA, as these choices yielded the best results among all publicly available word2vec, GloVe, SENNA (Collobert, 2011; Collobert et al., 2011) and RNNLM (Mikolov et al., 2011) word vectors.

The effects of the history sizes d_1 and d_2 for the short-text and the class representations, respectively, are presented in Table 3 for both the LSTM and CNN models. In both models, increasing d_1 while keeping $d_2 = 0$ improved the performances by 1.3-4.2 percentage points. Conversely, increasing d_2 while keeping $d_1 = 0$ yielded better results, but the performance increase was less pronounced: incorporating sequential information at the short-text representation level was more effective than at the class representation level.

Using sequential information at both the short-text representation level and the class representation level does not help in most cases and may even lower the performances. We hypothesize that short-text representations contain richer and more general information than class representations due to their larger dimension. Class representations may not convey any additional information over short-text representations, and are more likely to propagate errors from previous misclassifications.

Table 4 compares our results with the state-of-the-art. Overall, our model shows competitive results, while requiring no human-engineered features. Rigorous comparisons are challenging to draw, as many important details such as text preprocessing and train/valid/test split may vary, and many studies fail to perform several runs despite the randomness in some parts of the training process, such as weight initialization.

Model	DSTC 4	MRDA	SwDA
CNN	65.5	**84.6**	**73.1**
LSTM	**66.2**	84.3	69.6
Majority class	25.8	59.1	33.7
SVM	57.0	–	–
Graphical model	–	81.3	–
Naive Bayes	–	82.0	–
HMM	–	–	71.0
Memory-based Learning	–	–	72.3
Interlabeler agreement	–	–	84.0

Table 4: Accuracy (%) of our models and other methods from the literature. The majority class model predicts the most frequent class. SVM: (Dernoncourt et al., 2016). Graphical model: (Ji and Bilmes, 2006). Naive Bayes: (Lendvai and Geertzen, 2007). HMM: (Stolcke et al., 2000). Memory-based Learning: (Rotaru, 2002). All five models use features derived from transcribed words, as well as previous predicted dialog acts except for Naive Bayes. The interlabeler agreement could be obtained only for SwDA. For the CNN and LSTM models, the presented results are the test set accuracy of the run with the highest accuracy on the validation set.

5 Conclusion

In this article we have presented an ANN-based approach to sequential short-text classification. We demonstrate that adding sequential information improves the quality of the predictions, and the performance depends on what sequential information is used in the model. Our model achieves state-of-the-art results on three different datasets for dialog act prediction.

References

[Ang et al.2005] Jeremy Ang, Yang Liu, and Elizabeth Shriberg. 2005. Automatic dialog act segmentation and classification in multiparty meetings. In *ICASSP (1)*, pages 1061–1064.

[Blunsom et al.2014] Phil Blunsom, Edward Grefenstette, Nal Kalchbrenner, et al. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics.

[Cho et al.2014] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

[Collobert et al.2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

[Collobert2011] Ronan Collobert. 2011. Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics*, number EPFL-CONF-192374.

[Dernoncourt et al.2016] Franck Dernoncourt, Ji Young Lee, Trung H. Bui, and Hung H. Bui. 2016. Adobe-MIT submission to the DSTC 4 Spoken Language Understanding pilot task. In *7th International Workshop on Spoken Dialogue Systems (IWSDS)*.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

[Janin et al.2003] Adam Janin, Don Baron, Jane Edwards, Dan Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, et al. 2003. The ICSI meeting corpus. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 1, pages I–364. IEEE.

[Ji and Bilmes2006] Gang Ji and Jeff Bilmes. 2006. Backoff model training using partially observed data: application to dialog act tagging. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 280–287. Association for Computational Linguistics.

[Jurafsky et al.1997] Dan Jurafsky, Elizabeth Shriberg, and Debra Biasca. 1997. Switchboard SWBD-DAMSL shallow-discourse-function annotation coders manual. *Institute of Cognitive Science Technical Report*, pages 97–102.

[Kalchbrenner et al.2014] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.

[Kim et al.2015] Seokhwan Kim, Luis Fernando D'Haro, Rafael E. Banchs, Jason Williams, and Matthew Henderson. 2015. Dialog State Tracking Challenge 4: Handbook.

[Kim et al.2016] Seokhwan Kim, Luis Fernando D'Haro, Rafael E. Banchs, Jason Williams, and Matthew Henderson. 2016. The Fourth Dialog State Tracking Challenge. In *Proceedings of the 7th International Workshop on Spoken Dialogue Systems (IWSDS)*.

[Kim2014] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1746–1751. Association for Computational Linguistics.

[Lendvai and Geertzen2007] Piroska Lendvai and Jeroen Geertzen. 2007. Token-based chunking of turn-internal dialogue act sequences. In *Proceedings of the 8th SIGDIAL Workshop on Discourse and Dialogue*, pages 174–181.

[Mikolov et al.2011] Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Cernocky. 2011. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201.

[Mikolov et al.2013a] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[Mikolov et al.2013b] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[Nakagawa et al.2010] Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using CRFs with hidden variables. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 786–794. Association for Computational Linguistics.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12:1532–1543.

[Reithinger and Klesen1997] Norbert Reithinger and Martin Klesen. 1997. Dialogue act classification using language models. In *EuroSpeech*. Citeseer.

[Rotaru2002] Mihai Rotaru. 2002. Dialog act tagging using memory-based learning. *Term project, University of Pittsburgh*, pages 255–276.

[Schuster and Paliwal1997] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681.

[Shriberg et al.2004] Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. Technical report, DTIC Document.

[Silva et al.2011] Joao Silva, Luísa Coheur, Ana Cristina Mendes, and Andreas Wichert. 2011. From symbolic to sub-symbolic information in question classification. *Artificial Intelligence Review*, 35(2):137–154.

[Socher et al.2012] Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics.

[Stolcke et al.2000] Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

[Wang and Manning2012] Sida Wang and Christopher D Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics.

[Zeiler2012] Matthew D Zeiler. 2012. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.