Text-Based Speaker Identification For Multi-Participant Open-Domain Dialogue Systems

Iulian V. Serban *
Department of Computer Science and Operations Research,
Université de Montréal

Joelle Pineau †
School of Computer Science,
McGill University

Abstract

Understanding the interactive structure of dialogues, such as turn taking behaviour and change of speakers, is a critical prerequisite for dialogue systems which aim to understand and communicate using natural spoken language. In this work, we take a data-driven approach to learn turn changes and speaker identity from an open-domain corpus of movie scripts. We frame the problem as two distinct classification tasks and investigate the potential of different machine learning models, including logistic regression and recurrent neural networks. Our results indicate that it is feasible to estimate turn taking and speaker identity with high accuracy.

1 Introduction

Researchers in the fields of psychology and linguistics have long recognized the importance of studying the interactive structure of human dialogue, and have therefore spent considerable efforts to understand its components including non-verbal behaviour, prosody and utterances (Goodrich, 1979; Ford and Thompson, 1996; Dewaele and Furnham, 2000). Research on dialogue systems has also acknowledged the importance of these elements as a means for building more natural dialogue systems (Raux *et al.*, 2006) and, in particular, for systems which are able to take initiative themselves (Walker and Whittaker, 1995; Nakano *et al.*, 1999). Despite the impressive improvements in speech recognition and natural language understanding of recent years, most methods for detecting turn changes remain based on heuristic rules (Raux *et al.*, 2006; Gravano and Hirschberg, 2011). For example, a dialogue system would assume its interlocutor has finished his or her turn and begin its response once silence has been observed for a certain period of time (e.g. half of a second). However, such heuristic rules typically lead to unnatural and rigid dialogues (Raux *et al.*, 2006). Some researchers recently turned to estimating turn taking based on lexico-syntactic, prosodic and acoustic cues from human interlocutors(Gravano and Hirschberg, 2011; Bredin *et al.*, 2014). With the exception of Bredin *et al.* (2014), these approaches all focused on one-on-one task-driven dialogues, which are hard to transfer to multi-participant dialogues (dialogues with more than two interlocutors), and to more open-topic conversations. These approaches were all based on prosodic and acoustic cues, which require an audio stream together with a speech recognition software or classifier that outputs prosodic cues.

In a related line of work, research on speech recognition has attacked the problem of speaker identification (Miro *et al.*, 2012; Tranter *et al.*, 2006). In the speech recognition community, this is considered a subtask of the speaker diarisation task. Largely based on acoustic signals, i.e. characteristics of speaker voices, researchers have built highly accurate models for multi-participant dialogues. However, it is commonly acknowledged that natural language text itself reflects the personality of the speaker, in addition to its semantic content (Mairesse *et al.*, 2007). This leads us to the interesting open problem of whether or not algorithms are able to distinguish speakers based only on

*http://www.iulianserban.com
†http://cs.mcgill.ca/~jpineau/

text and context. This problem is further motivated by the increasing number of large unstructured dialogue datasets, such as the Ubuntu Dialogue Corpus (Lowe *et al.*, 2015) and OpenSubtitles Corpus (Tiedemann, 2012) consisting only of the dialogue text, where no speaker labels are available. Building accurate speaker identification models allows estimating speaker labels in such corpora.

In this work, we take a step in this direction by designing data-driven models for turn taking and speaker identification using data from open-domain multi-participant dialogues. To train our models, we leverage a publicly available text corpus based on movie scripts. We focus on probabilistic models, which can be extended with additional contextual, acoustic and prosodic signals.

2 Classification Tasks

We define two classification tasks. The first is a turn taking binary classification task. Given two consecutive sentences, the model must classify the sentences as *turn-change*, i.e. the speaker of the first sentence is different from the speaker of the second sentence, or *no-turn-change*, i.e. the speaker is the same for both sentences.

The second task is a 6-way classification task, which we will refer to as the speaker identification task. Given two consecutive sentences, the model must classify the sentences as one of six classes:

- Class 1: a single speaker without any pause,
- Class 2: a single speaker with a pause,
- Class 3: two speakers, where second speaker is <first_speaker>,
- Class 4: two speakers, where second speaker is <second_speaker>,
- Class 5: two speakers, where second speaker is <third_speaker>,
- Class 6: two speakers, where second speaker is <minor_speaker>.

The task can trivially be expanded to include additional speakers, but due to data scarcity we will restrict ourselves to six classes. The task can be converted into the binary turn taking classification task by redefining classes 1)-2) as the *no-turn-change* class and classes 3)-6) as the *turn-change* class.

For both tasks, the model is also allowed to condition its prediction on all previous sentences in the script, but without knowing the true speakers.

3 Dataset

We use the *Movie-Scriptolog* dataset (Serban *et al.*, 2015) consisting of 614 movie scripts, which was developed by expanding and preprocessing another movie script corpus (Banchs, 2012) based on *The Internet Movie Script Database*[1]. We choose to work with films because they reflect natural spoken interactions between humans (Forchini, 2009). For example, based on analysing a corpus of a hundred transcribed films, Forchini (2009) observes that: *"movie language can be regarded as a potential source for teaching and learning spoken language features"*. Although movie scripts are written by manuscript writers, they are subsequently edited by producers and actors and therefore reflect a high degree of realism. This is especially the case for our corpus, because the majority of films we include are very well-known; 522 of the films have more than 10,000 votes on IMDB[2].

The movie scripts in the dataset were preprocessed using regex expressions and spell checking and afterwards tokenized with the Moses tokenizer developed by Josh Schroeder[3]. Furthermore, due to the differences in punctuation and casing style between scripts, all tokens were lower-cased and each punctuation mark was made into a separate token. Each manuscript is a separate entity consisting of a sequence of tokens. Each utterance ends with an end-of-utterance token </s>, and starts with a *speaker token*: <first_speaker>, <second_speaker>, <third_speaker>, <minor_speaker>, which

[1] www.imsdb.com

[2] www.imdb.com

[3] github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl

respectively represents the most frequent speaker of the movie (e.g. main actor), the second most frequent speaker (e.g. supporting actor), the third most frequent speaker and all other speakers, i.e. all less frequent speakers are given the label <minor_speaker>. Furthermore, for voice over and off screen utterances respectively, the speaker token is followed by one of the two *auxiliary tokens* <voice_over> and <off_screen>. The corpus also contains a special *pause token* denoted <pause>, which is placed between consecutive, uninterrupted utterances of the same speaker.

For the classification tasks described above, we break each script in the *Movie-Scriptolog* dataset into a sequence of sentences always separated by either punctuation marks, question marks or exclamation marks. We then remove all speaker tokens and auxiliary token from each sentence. We keep the training, validation and test set splits from the *Movie-Scriptolog* dataset. Statistics for each class in the two tasks are outlined in Table 1. The dataset will be made available upon request.

Turn Taking Task	Speaker Classification Task	Training	Validation	Test
No-Turn-Change	Class 1	416,264	55,157	58,866
No-Turn-Change	Class 2	837	107	97
Turn-Change	Class 3	93,135	12,643	12,171
Turn-Change	Class 4	53,455	7,266	7,087
Turn-Change	Class 5	31,670	4,375	4,142
Turn-Change	Class 6	118,858	16,352	16,460

Table 1: Label instances for the two classification tasks.

4 Models

Our objective is to build an automatic classifier for the two tasks defined above. We consider two baseline models, then experiment with a standard recurrent neural network (RNN model), and finally explore variants that condition on additional information from the conversation.

4.1 Baseline models

The first baseline model always predicts the most frequent class. We call this model the *Majority Class Predictor*. This is the most naive baseline available and useful simply to define the difficulty of the task.

The second baseline model is a logistic regression model, which takes as input a set of hand-crafted features. Largely inspired by the work of Walker *et al.* (2012), we compute the following features for each of the two sentences:

- Sum over word sentiment polarities using SentiWordNet (Esuli and Sebastiani, 2006).
- Number of *bad words* (e.g. swear words)[4].
- Probabilities of sentence belonging to one of 15 dialogue act types according to a naive Bayes classifier. The naive Bayes classifier was based on bag-of-words features and trained with maximum likelihood on the NPS corpus using NLTK[5].
- Sum of TF-IDF (term frequency times inverse document frequency) values for each word in the sentence, where the document is defined to be all the words in the movie script.

We computed additional features as transformations of these by subtracting the features of the first sentence from that of the second sentence, as well as by taking their absolute values. This yielded 80 features in total as input to the logistic regression model, which was then trained by optimizing the log-likelihood separately for each task. We call this the *Logistic Regression* model.

[4]Based on the following word list: `github.com/shutterstock/List-of-Dirty-Naughty-Obscene-and-Otherwise-Bad-Words/blob/master/en`

[5]NLTK (Natural Language Toolkit) is a python natural language processing library, which can be downloaded at `www.nltk.org`. Further details on the naive Bayes classifier is described at `www.nltk.org/book/ch06.html`.

4.2 Recurrent Neural Networks

We propose to use recurrent neural networks (RNNs) for embedding the words before and after the classification labels into a distributed vector space representation (Bengio *et al.*, 2003; Mikolov *et al.*, 2010). It might be useful to embed more context, hence we will embed the previous t tokens before the classification label, and the next t tokens after the classification label. Inspired by the approach taken by Yu *et al.* (2014) and Lowe *et al.* (2015), we use two GRU RNNs (Cho *et al.*, 2014) to transform the words into a vector representation. The first RNN reads the previous t tokens word-by-word forwards producing a sequence of hidden states $h_1^p, \ldots, h_t^p$, where p stands for *previous*. The last hidden state of the RNN is taken to be the vector representation of the previous sentences: $v^p = (h_t^p)^T$. Likewise, the second RNN reads the next t tokens word-by-word forwards producing another sequence of hidden states $h_1^n, \ldots, h_t^n$, where n stands for *next*. Its last hidden state is taken to be the vector representation of the next sentences: $v^n = h_t^n$. We constrain the RNNs to share parameters for the word embeddings. We assume that both RNNs have the same hidden state size q, which implies that the sentence embeddings also have dimensionality q: $v^p, v^n \in \mathbb{R}^q$. Further details on the RNN architectures are given by Cho *et al.* (2014).

In addition to the sentence embeddings, we also want to condition the models on the hand-crafted features discussed above. Let $F \in \mathbb{R}^{80}$ be the vector of 80 hand-crafted features. Let $M_i \in \mathbb{R}^{q \times q}$ and $L_i \in \mathbb{R}^{80}$ respectively be parameter matrices and vectors for $i = 1, \ldots, k$, where k is the number of classes defined by the classification task. The model assigns probabilities defined by:

$$P_\theta(\text{speaker class = c} \mid \text{sentences}) = \frac{\exp(v^p M_c v^n + L_c^T F)}{\sum_i \exp(v^p M_i v^n + L_i^T F)}, \tag{1}$$

which, for fixed v^p and v^n, can be interpreted as a logistic regression model conditioned on both the sentence embeddings and the hand-crafted features. The model is illustrated in Figure 1.

To help generalization we regularize the model. First, we restrict M_i, for classes $i = 1, \ldots, k$, to be diagonal. We then rewrite the term inside the exponential function of the numerator in eq. (1):

$$v^p M_i v^n = \text{Tr}(v^p M_i v^n) = \text{Tr}(M_i v^n v^p) = \sum_j M_{i,jj} v_j^n v_j^p = \sum_j M_{i,jj} (v^n \cdot v^p)_j,$$

where $\cdot$ is the Hadamard product. The last equality decomposes the result into a weighted sum over the elements of $v^n \cdot v^p$. We can therefore regularize the model further by linearly projecting $v^n \cdot v^p$ to a new vector $v_{\text{low-dimension}} \in \mathbb{R}^r$ of low dimensionality, s.t. $v_{\text{low-dimension}} = O(v^n \cdot v^p)$, where $O \in \mathbb{R}^{q \times r}$ is a parameter matrix and $r < q$. This is analogous to performing dimensionality reduction in the space of the product of the embeddings, and reduces the number of parameters for $i = 1, \ldots, k$ in the diagonal matrices $M_i \in \mathbb{R}^{r \times r}$. We call this the ***RNN*** model.

To help generalization further, we may fix the word embedding parameters with Word2Vec word embeddings[6] (Mikolov *et al.*, 2013) trained on the *Google News* dataset containing about 100 billion words. This should improve the semantic representation of each word, since the *Google News* dataset is more than four orders of a magnitude larger than the *Movie-Scriptolog* dataset. We call this the ***RNN+Word2Vec*** model.

We train both models by optimizing the log-likelihood of the labels, where all parameters are shared between the models for each task with the exception of M_i, L_i for $i = 1, \ldots, k$, which are estimated separately for each task[7].

4.3 Conditioning On The Previous Speaker

To take earlier context into account, we may consider the sequence of speakers to be a Markov chain, where the current speaker depends on only the previous speaker and the utterance of the

[6] `code.google.com/p/word2vec/`

[7] The actual optimization criterion used in our experiments was a sum of three log-likelihoods: the log-likelihood of the binary turn taking label, the log-likelihood of the speaker classification label and the log-likelihood of the auxiliary token (<voice_over> <off_screen> or none). Separate parameters M_i, L_i for $i = 1, \ldots, k$ were optimized for predicting the auxiliary tokens. However, our experiments showed that adding the last log-likelihood term did not affect the classification accuracies of the two tasks, but it does enable the model to predict whether there is a <voice_over> or <off_screen> token in the second sentence, which might be useful for certain applications.

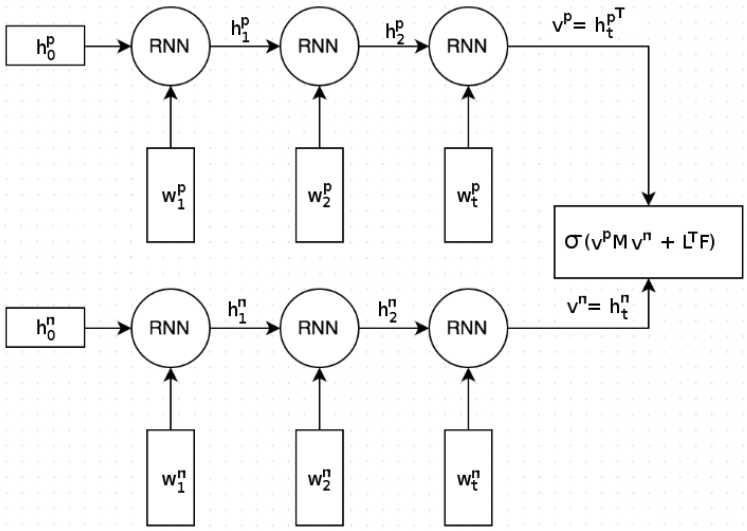

Figure 1: The computational graph of the RNN model. The vector embeddings v^p and v^n are produced respectively by running one RNN over the word sequences in the previous and preceding sentences and one RNN over the word sequences in the next and proceeding sentences. σ represents the softmax function as defined in eq. (1). Adapted from Lowe *et al.* (2015).

.

previous speaker. Formally, let s_n be the discrete speaker class at turn n as defined for the speaker classification task. Let u_n be the sentence at turn n, and assume that there are N sentences. Now we can define the following directed graphical model over speakers and sentences:

$$P(s_1, u_1, \ldots, s_N, u_N) = P(s_1)P(u_1|s_1) \prod_{n=2}^{N} P(s_n|s_{n-1}, u_{n-1})P(u_n|s_n), \tag{2}$$

where we assume the transitions follow a time-homogeneous Markov chain. However, we are interested in modelling the distribution over speakers. Therefore, we use Bayes rule to compute the posterior:

$$P(s_n|u_n, s_{n-1}, u_{n-1}) \propto P(s_n, u_n|s_{n-1}, u_{n-1}) = P(s_n|s_{n-1}, u_{n-1})P(u_n|s_n) \tag{3}$$

This motivates our last model, which similarly is conditioned on the previous speaker as well as the current and previous sentences. This model replaces eq. (1) with parameters conditioned on the previous speaker class:

$$\begin{aligned} &P_\theta(\text{speaker class = c} \mid \text{sentences, previous speaker class } = c_{\text{prev}}) \\ &\qquad = \frac{\exp(v^p M_c^{c_{\text{prev}}} v^n + \left(L_c^{c_{\text{prev}}}\right)^T F)}{\sum_i \exp(v^p M_i^{c_{\text{prev}}} v^n + \left(L_i^{c_{\text{prev}}}\right)^T F)}, \end{aligned} \tag{4}$$

where c_{prev} is the class of the previous speaker as defined in the speaker classification task. As before, we can train this model by optimizing the log-likelihood of the labels. At test time, we classify the speakers in each script chronologically: for each sentence we estimate the speaker as the class with the highest probability under the model, and assume that this is the true label to condition on when estimating the speaker of the next sentence. Furthermore, we assume that the speaker of the very first sentence in each script was preceded by the <minor_speaker> (Class 6) class. We call this the *Conditioned RNN* model.

This extension can also be applied to the Logistic Regression model. In this case, we train a separate logistic regression model for each previous speaker label, and evaluate it similarly to the Conditioned RNN model. We call this model the *Conditioned Logistic Regression* model.

Model	Turn Taking Task	Speaker Classification Task
Majority Class Predictor	59.67%	59.57%
Logistic Regression	63.70%	59.25%
Conditioned Logistic Regression	62.90%	59.48%
RNN	88.85%	68.92%
RNN+Word2Vec	**89.47%**	**69.47%**
Conditioned RNN	88.99%	68.76%

Table 2: Test classification accuracies.

4.4 Results

We optimize all models using the first-order stochastic gradient optimization method Adam (Kingma and Ba, 2015)[8]. We choose our hyperparameters by early stopping with patience on the validation set log-likelihood (Bengio, 2012). For the RNN-based models, we experiment with parameters $q \in \{50, 100, 200\}$, $t \in \{10, 15, 20, 25, 30\}$, $r = 50$ and word embeddings dimensionality of size 50 and 100. For the RNN+Word2Vec model, we project the 300 dimensional Word2Vec embeddings learned from the *Google News* dataset down to word embeddings of dimensionality $t = 50$ using principal component analysis. The parameters of the Conditioned RNN and Conditioned Logistic Regression models were initialized from respectively the parameters of the RNN and Logistic Regression models.

The results are given in Table 2. First, we observe that the Logistic Regression is able to outperform the Majority Class baseline on the turn taking task but not on the speaker classification task. Second, we see that the RNN-based models clearly outperform the Logistic Regression model on both classification tasks. This suggests that the RNNs have learned sentence-level embeddings that capture speaker discriminative lexico-syntatic and semantic cues. In particular, the observation that the RNN+Word2Vec model performs slightly better than the RNN model suggests that additional labelled training data may improve performance substantially. However, contrary to our expectations, neither the Conditioned Logistic Regression nor the Conditioned RNN models performed substantially better than their unconditioned counterparts. This might suggest that the features computed by these models implicitly already take into account the previous speaker class, or it might suggest that the previous speaker class does not provide discriminative information for classifying the current speaker class. Further investigation is necessary.

5 Discussion

This work presents a data-driven approach to automatically infer turns and speakers from scripted dialogues. The models considered show promising performances for automatically identifying turn taking and speaker cues in multi-participant open-domain dialogues. They allow rich probabilistic inference, and could be further augmented to consider more speaker characteristics, such as features based on audio and task context. Future work should apply these to spoken dialogue corpora.

In related work, Walker *et al.* (2012) used movie scripts to classify the personality of characters. They showed that film characters were often based on stereotypical roles, and that these could be distinguished according to the manuscript writer and the film genre, suggesting that movie scripts may serve as a test bed for experiments related to personality characterization. Clearly, the possibility of inferring additional information about speakers holds significant potential for personalizing the delivery of services, recommendations, advertising, and much more. Our results suggest that recurrent neural networks offer a rich paradigm for building models for speaker characterization.

Acknowledgments

The authors acknowledge IBM Research, NSERC, Canada Research Chairs, CIFAR and Compute Canada for funding. The authors thank Caglar Gulcehre, Ryan Lowe and Laurent Charlin for constructive feedback.

[8]The source code is available on `github.com/julianser/dlg-segmenter`

References

Banchs, R. E. (2012). Movie-dic: a movie dialogue corpus for research and development. In *Proceedings of the Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 203–207.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*, pages 437–478. Springer.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, **3**, 1137–1155.

Bredin, H., Roy, A., Pécheux, N., and Allauzen, A. (2014). Sheldon speaking, bonjour!: Leveraging multilingual tracks for (weakly) supervised speaker identification. In *Proceedings of the ACM International Conference on Multimedia*, pages 137–146. ACM.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734.

Dewaele, J.-M. and Furnham, A. (2000). Personality and speech production: a pilot study of second language learners. *Personality and Individual Differences*, **28**(2), 355–365.

Esuli, A. and Sebastiani, F. (2006). Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422.

Forchini, P. (2009). Spontaneity reloaded: American face-to-face and movie conversation compared. In *Abstracts The 5th Corpus Linguistics Conference*.

Ford, C. E. and Thompson, S. A. (1996). Interactional units in conversation: Syntactic, intonational, and pragmatic resources for the management of turns. *Studies in interactional sociolinguistics*, **13**, 134–184.

Goodrich, W. (1979). Face-to-face interaction: Research, methods, and theory. *Family Process*, **18**(3), 355–356.

Gravano, A. and Hirschberg, J. (2011). Turn-taking cues in task-oriented dialogue. *Computer Speech & Language*, **25**(3), 601–634.

Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Lowe, R., Pow, N., Serban, I., and Pineau, J. (2015). The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the SIGDIAL 2015 Conference*. In press.

Mairesse, F., Walker, M., Mehl, M., and Moore, R. (2007). Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research*, pages 457–500.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Conference of the International Speech Communication Association*, pages 1045–1048.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Miro, X. A., Bozonnet, S., Evans, N., Fredouille, C., Friedland, G., and Vinyals, O. (2012). Speaker diarization: A review of recent research. *IEEE Transactions on Audio, Speech, and Language Processing*, **20**(2), 356–370.

Nakano, M., Dohsaka, K., Miyazaki, N., Hirasawa, J.-i., Tamoto, M., Kawamori, M., Sugiyama, A., and Kawabata, T. (1999). Handling rich turn-taking in spoken dialogue systems. In *EUROSPEECH*.

Raux, A., Bohus, D., Langner, B., Black, A. W., and Eskenazi, M. (2006). Doing research on a deployed spoken dialogue system: one year of let's go! experience. In *INTERSPEECH*.

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. (2015). Hierarchical Neural Network Generative Models for Movie Dialogues. *ArXiv e-prints*. 1507.04808.

Tiedemann, J. (2012). Parallel data, tools and interfaces in opus. In *Proceedings of LREC*, pages 2214–2218.

Tranter, S. E., Reynolds, D., *et al.* (2006). An overview of automatic speaker diarization systems. *IEEE Transactions on Audio, Speech, and Language Processing*, **14**(5), 1557–1565.

Walker, M. and Whittaker, S. (1995). Mixed Initiative in Dialogue: An Investigation into Discourse Segmentation. In *Proceedings of the Meeting on Association for Computational Linguistics*, pages 70–78.

Walker, M. A., Lin, G. I., and Sawyer, J. (2012). An annotated corpus of film dialogue for learning and characterizing character style. In *Proceedings of LREC*, pages 1373–1378.

Yu, L., Hermann, K. M., Blunsom, P., and Pulman, S. (2014). Deep learning for answer sentence selection. In *Workshop on Deep Learning, Advances in Neural Information Processing Systems*.