

# Positional Transformers for Claim Span Identification

Michael Sullivan<sup>1</sup>, Navid Madani<sup>1</sup>, Sougata Saha<sup>1</sup> and Rohini Srihari<sup>1</sup>

<sup>1</sup>University at Buffalo, Buffalo, NY 14260, United States

## Abstract

Given the vast amount of misinformation present in today's social media environment, it is critical to be able to identify claims made in social media posts to facilitate the fact-verification process. For this reason, the CLAIMSCAN (Task B) shared task introduces the objective of *claim span identification*, which requires identifying spans of text within tweets that correspond to (allegedly) factual claims made by users. In this submission to CLAIMSCAN Task B, we introduce the *positional transformer* architecture for claim span identification. This architecture utilizes a novel, position-sensitive attention mechanism that is able to outperform all other submissions to the shared task, but still falls behind a few of the task organizers' more complex baseline models. In this paper, we discuss the positional transformer architecture, the training and data pre-processing procedures used for CLAIMSCAN Task B, and our results on this task.

## Keywords

CLAIMSCAN, Claim span identification, Social media, Transformers

## 1. Introduction

Today's online social media users are inundated with various claims about current (and past) events, hindering their ability to discern fact from fiction. To combat this deluge of (mis-)information, and better equip users to filter out false claims online, Task B of the CLAIMSCAN [1] shared task requires systems to identify spans in tweets corresponding to *claims* ("assertion[s] that deserve[...] our attention" [2] or "argumentative component[s] in which the speaker or writer conveys the central, contentious conclusion of their argument" [3]), using the *Claim Unit Recognition in Tweets* (CURT) dataset. The ultimate goal of this line of research is to "empower the fact checkers"; in other words, to facilitate fact-checking in tweets by flagging only those parts of the text that require verification.

In this submission to CLAIMSCAN Task B, we introduce the *positional transformer* architecture, a modified variant of the transformer encoder [4] architecture that utilizes a position-sensitive attention mechanism (*positional attention*). We find that the positional transformer outperforms all other submissions to this task, with the exception of some of the organizer's baseline models, one of which employs gated, additive/subtractive attention between the input text and a series of hand-crafted templates describing the content and/or structure that a given claim may take. This suggests that, in the absence of the requisite time or resources needed to

---

Forum for Information Retrieval Evaluation, December 15-18, 2023, India


✉ mjs227@buffalo.edu (M. Sullivan); smadani@buffalo.edu (N. Madani); sougatas@buffalo.edu (S. Saha); rohini@buffalo.edu (R. Srihari)

🌐 <https://www.acsu.buffalo.edu/~mjs227> (M. Sullivan); <https://sougata-ub.github.io/> (S. Saha);

<https://www.acsu.buffalo.edu/~rohini/> (R. Srihari)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

Some young people still think only older folks die when they get Coronavirus and younger people are <u>somehow immune to it</u> . Young people have also died with Covid19 and some face health complications that could last them for their lifetime #COVID19 #COVIDIOTS Stay At Home Save Lives
@micaela_ayye I am only worried about it bc <u>there is no cure/ vaccine</u> . Atm <u>the only the only thing we can do is treatment of symptoms</u> , so keeping them hydrated and seeing whether or not their bodies kill it. #coronavirus
<u>just in three tsa officers at sanjose intl airport have tested positive for coronavirus</u> tsa all tsa employees they have come in contact with them over the past 14 days are quarantined at home sjc airtravel full statement
How about <u>China pick up the worlds dead bodies that they killed with their bio-weapon</u> #coronavirus

**Table 1**

Examples of tweets from the CURT dataset. Claim spans are underlined.

hand-craft such templates, the positional transformer may present one of the current optimal architectures for claim span identification.

## 2. Task and Dataset Description

The CURT dataset consists of 6044, 756, and 755 tweets in the train, development, and test sets, respectively. Each tweet is pre-segmented into a list of “tokens”; note that (perhaps obviously) these “tokens” do not necessarily align with the tokens generated by a given language model’s tokenizer, and some of the “tokens” in the token lists provided in the dataset may be segmented into two or more tokens by the LM’s tokenizer. Claim spans are then given as pairs of list indices indicating the start/end tokens of each span. See Table 1 for examples of tweets from the dataset and claim spans within them.

The vast majority of the tweets in the dataset contain at least one claim span, and some contain two or more. A small fraction—approximately 0.4%—of the tweets do not contain any claim spans. On average, approximately 56% of the tokens in each tweet in the CURT dataset are contained within a claim span. See Sundriyal et al. [1] for more details on the construction of and statistics regarding this dataset.

The evaluation metric for this task is the token-level F1 score, averaged over each tweet in the dataset.

### 3. Related Work

The organizers of the CLAIMSCAN task introduce the *DaBERTa* (Description Aware RoBERTa; see Sundriyal et al. [1]) architecture as a baseline for the claim span identification task. We refer interested readers to their work for a detailed description of the DaBERTa architecture, but provide a brief overview below.

DaBERTa consists of the RoBERTa-base model [5] coupled with a *Description Infuser Network* (DescNet) classification head. First, the claim description templates  $d_i$ —hand-crafted templates describing the content and/or structure that a given claim may take—are passed through a pre-trained RoBERTa model to obtain sentence embeddings  $d'_i$ . Then, each input tweet  $t_j$  is passed through another pre-trained RoBERTa model, and a *Compositional De-Attention* (CoDA; [6]) block generates a representation  $z_j$  of the claim description embeddings  $d'_i$ , attention-weighted by the input tweet tokens  $t_{jk}$ . The claim description representations  $z_j$ , along with the RoBERTa-encoded tweet text  $t_j$ , are then passed to the *Interactive Gating Mechanism* (IGM); a series of pointwise-multiplicative gates (c.f. output gates in LSTM [7] architectures), which aims to capture semantically similar and dissimilar features between  $z_j$  and  $t_j$ . Finally, the output of the IGM is passed to a *Conditional Random Field* (CRF; [8]) layer to obtain predicted labels for each of the tokens in  $t_j$ . This entire architecture is then trained end-to-end. Furthermore, the token target labels are *Beginning-Inside-Outside* (BIO; [9]) encoded: tokens occurring at the beginning of a claim span, tokens within a claim span not contained within the previous category, and tokens not contained within a claim span each receive separate respective labels.

The DaBERTa architecture achieves a token-level F1 score of **0.8604** on the CURT dataset (see Table 2). This represents the current state-of-the-art for claim span detection, which is a novel task first introduced in the CLAIMSCAN competition.

Due to its novelty, there (perhaps obviously) does not exist any prior work on claim span detection, aside from that of Sundriyal et al. [1]. However, claim span detection is closely related to the field of *Argument Mining* (AM), which involves identifying arguments in text and relations between them [10]. As such, we briefly discuss related work from this area.

In their work on AM, Chakrabarty et al. [11] utilize a Rhetorical Structure Theory [12] parser (for feature extraction) coupled with a fine-tuned BERT model [13] to identify argument spans in online discussions. The authors find that this approach yields high recall, but low precision, on AM tasks. Similarly, Cheng et al. [14] feed pre-trained BERT embeddings to a bidirectional LSTM [15] classification head to detect reviewers’ objections and authors’ rebuttals in data gathered from the peer-review process.

### 4. Approach Description

In this section, we first outline the *positional transformer* architecture (Section 4.1). We then discuss the data preprocessing procedures and training setup/hyperparameters utilized in this submission to the CLAIMSCAN task (Section 4.2).<sup>1</sup>

---

<sup>1</sup>All code available on GitHub: <https://github.com/mjs227/CLAIMSCAN>

#### 4.1. Positional Transformer

As mentioned in Section 1 above, the positional transformer is a modified variant of the transformer encoder [4] architecture that utilizes a position-sensitive attention mechanism that we refer to as *positional attention*. As with the DescNet and BiLSTM components of the approaches of Sundriyal et al. [1] and Cheng et al. [14] (respectively) discussed in Section 3 above, the positional transformer is merely a classification head designed for sequence classification, and must be strapped on top of an embedding model—following Sundriyal et al. [1], we use RoBERTa-base [5] as the underlying language model for this task.

Given an input text  $t$  of length  $N$  (in tokens), the positional transformer acts on each token individually, while taking into account the tokens surrounding it. First,  $t$  is passed to the RoBERTa model to obtain a sequence of embeddings. The sequence of embeddings of dimension  $d_{\text{RoBERTa}}$  are then downsampled via a linear layer to the significantly smaller positional transformer embedding dimension ( $d_{PT}$ ), to obtain a sequence of  $d_{PT}$ -dimensional embeddings  $t'$ . Then, we construct the *window*  $W(t') \in \mathbb{R}^{N \times (S_L^W + S_R^W + 1) \times d_{PT}}$ , where for each  $1 \leq i \leq N$ ,  $W(t')_i \in \mathbb{R}^{(S_L^W + S_R^W + 1) \times d_{PT}}$  is centered on the  $i^{\text{th}}$  token embedding  $t'_i$ . The window  $W(t')_i$  consists of the  $S_L^W$  and  $S_R^W$  tokens preceding and following  $t'_i$  (Equation 1), where  $S_L^W$  and  $S_R^W$  are hyperparameters denoting the left- and right-hand window sizes, respectively.

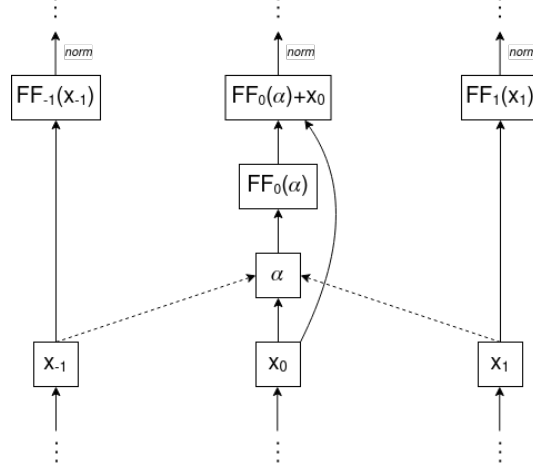
$$W(t'')_i = \text{Concat}(t''_{i-S_L^W}, \dots, t''_{i-1}, t''_i, t''_{i+1}, \dots, t''_{i+S_R^W}) \quad (1)$$

To account for those tokens  $t'_i$  near the beginning and end of the sequence such that  $i - S_L^W < 1$  or  $i + S_R^W > N$ , we define  $t''$  (for *all* integers  $k$ ) as follows in Equation 2 below.

$$t''_k = \begin{cases} t'_0 & \text{if } k < 1 \\ t'_{N+1} & \text{if } k > N \\ t'_k & \text{otherwise} \end{cases} \quad (2)$$

Where  $t'_0$  and  $t'_{N+1}$  denote the embeddings of the BOS and EOS tokens, respectively. For example, the window corresponding to the first *lexical* (i.e. non-EOS/BOS) token in the sequence,  $W(t')_1$ , consists of  $S_L^W$  copies of the BOS token embedding ( $t'_0$ ) concatenated with  $t'_{1:S_R^W+1}$  ( $t'_1$ , along with the following  $S_R^W$  token embeddings in  $t'$ ). On the other hand, the window corresponding to the *last* lexical token in the sequence,  $W(t')_N$ , consists of  $t'_{N-S_L^W:N}$  ( $t'_N$ , along with the  $S_L^W$  preceding token embeddings in  $t'$ ) concatenated with  $S_R^W$  copies of the EOS token embedding ( $t'_{N+1}$ ).

Then, each window  $W(t')_i$  is passed to the positional transformer itself. The positional transformer consists of  $M$  *positional transformer layers*  $\{L_k\}_{1 \leq k \leq M}$  along with a single *positional transformer final layer*  $L_F$ . Each of the non-final layers are architecturally identical (see Figure 1), and consist of  $S_L^W + S_R^W + 1$  linear layers  $\{FF_j^k\}_{S_L^W \leq j \leq S_R^W}$ , along with the *positional attention block*. Positional attention is similar to dot-product attention as in Vaswani et al. [4], but restricted to each window  $W_i$  and focused solely on the *center* of  $W_i$  (i.e. the  $i^{\text{th}}$  token  $t_i$ ). As such, there are a few critical differences between “classical” dot-product attention and positional attention; namely, there is only one query vector (corresponding to the center  $t_i$ ), and a unique key projection matrix for each of the  $S_L^W + S_R^W + 1$  positions in the input window.



**Figure 1:** Architecture of a (non-final) positional transformer layer. In this example,  $S_L^W = S_R^W = 1$  for readability. The input  $x_0 = (W_i^{k-1})_0$  is the center (i.e. corresponds to the token  $t_i$ ) of the window  $W_i^{k-1}$ ;  $x_{-1} = (W_i^{k-1})_{-1}$  and  $x_1 = (W_i^{k-1})_1$  belong to the left- and right-hand contexts of  $W_i^{k-1}$  (respectively), and correspond to the tokens immediately to the left/right of  $t_i$ .

The motivation behind the separate feed-forward layers and key projection matrices in each positional transformer layer is to model the unique contributions that each position in the input window makes towards the prediction of the label for the center  $t_i$ . For example, if a token to the left of  $t_i$  belongs to a claim span and represents the beginning of a clause, that may increase the likelihood that  $t_i$  belongs to a claim span. However, if such a token occurs to the *right* of  $t_i$ , that may *decrease* the likelihood that  $t_i$  belongs to a claim span. This is conceptually similar to *disentangled attention* (e.g. DeBERTa; [16]), but, rather than implementing disentanglement via separate *embeddings* for position and content, the positional transformer uses separate *matrices* for each position.

For each  $1 \leq i \leq N$  and each  $0 \leq k \leq M$ , let  $W_i^k$  denote the output of the  $k^{\text{th}}$  non-final layer applied to the  $i^{\text{th}}$  window—i.e.  $W_i^0 = W_i$ , and for all  $1 \leq k' \leq M$ ,  $W_i^{k'} = L_{k'}(W_i^{k'-1})$ . Now, let  $(W_i^k)_0$  denote the embedding corresponding to  $t_i$  (the center of  $W_i$ ), and for each  $1 \leq j_L \leq S_L^W$  and  $1 \leq j_R \leq S_R^W$ , let  $(W_i^k)_{-j_L}$  and  $(W_i^k)_{j_R}$  denote the token embeddings  $j_L$  positions to the left and  $j_R$  positions to the right of the center  $(W_i^k)_0$ , respectively.

Within the positional attention block of the  $k^{\text{th}}$  layer, there are  $S_L^W$  *left-hand* key-projection (linear) layers  $\{K_j^k\}_{S_L^W \leq j < 0}$ ,  $S_R^W$  *right-hand* key-projection layers  $\{K_j^k\}_{0 < j \leq S_R^W}$ , and a single *central* key-projection layer  $K_0^k$ , along with a single query-projection layer  $Q^k$ . For each embedding  $(W_i^{k-1})_{-j_L}$  in the left-hand window, its corresponding key vector is defined to be  $K_{-j_L}^k((W_i^{k-1})_{-j_L})$ . Similarly, for each embedding  $(W_i^{k-1})_{j_R}$  in the *right-hand* window, its corresponding key vector is defined to be  $K_{j_R}^k((W_i^{k-1})_{j_R})$ . The central input,  $(W_i^{k-1})_0$ , corresponds to the key vector  $K_0^k((W_i^{k-1})_0)$  and the query vector  $Q^k((W_i^{k-1})_0)$ . Finally, we compute the attention weights  $A^k$  by taking the softmax of the dot products of each key vector with the single query vector (Equation 3).

$$\text{for all } -S_L^W \leq j \leq S_R^W : (A')_j^k = K_j^k((W_i^k)_j) \cdot Q^k((W_i^k)_0) \quad (3a)$$

$$A^k = \text{softmax}((A')^k) \quad (3b)$$

Due to the low values of  $d_{pT}$  used for this task, we do not normalize the attention values by dividing by the square root of the key dimension as in Vaswani et al. [4]. We then obtain an attention-weighted representation for the central input,  $\alpha^k$ , as in “classical” dot-product attention—i.e. as the attention-weighted sum of each embedding within the window (see Equation 4).

$$\alpha^k = \text{Norm}_{L2} \left( \sum_{j=-S_L^W}^{S_R^W} A_j^k (W_i^k)_j \right) \quad (4)$$

The central input,  $\alpha^k$ , is then passed through the central feed-forward (linear) layer  $FF_0^k$ , an additive skip connection, and a second  $L2$ -normalization, before being outputted by the layer  $L_{k+1}$ . For all  $-S_L^W \leq j \leq S_R^W$  such that  $j \neq 0$ ,  $(W_i^{k-1})_j$  is passed through the (unique)  $j^{\text{th}}$  feed-forward (linear) layer  $FF_j^k$  and  $L2$ -normalized (see Equation 5).

$$(W_i^k)_j = \begin{cases} \text{Norm}_{L2}(FF_0^k(\alpha^k)) & \text{if } j = 0 \\ \text{Norm}_{L2}(FF_j^k((W_i^{k-1})_j)) & \text{otherwise} \end{cases} \quad (5)$$

After passing through the first  $M$  non-final layers, the input  $W_i^M$  is then passed to the final layer  $L_F$ , which consists solely of a positional attention block. The output of this attention block—an attention-weighted representation of the tokens surrounding the central input  $t_i$ —is the output of the positional transformer.

Unlike Sundriyal et al.[1], we do not employ BIO encoding for the token labels, as we found that it decreased performance (F1 score) on the development set. Rather, we utilize a simpler, binary (“inside/outside”) labeling scheme. As such, the  $d_{pT}$ -dimensional output of the positional transformer with respect to the input  $W_i$  is then passed to a  $d_{pT} \times 1$  sigmoid layer to obtain a predicted label for the  $i^{\text{th}}$  token.

## 4.2. Data Preprocessing and Training

For the claim span detection task on the CURT dataset, utilized a three-layer positional transformer (four layers total, including the final layer) with  $d_{pT} = 5$  and  $S_L^W = S_R^W = 7$ . The entire RoBERTa  $\rightarrow$  Positional Transformer pipeline was trained end-to-end using token-level binary cross-entropy loss and the Adam [17] optimizer, with learning rates of  $10^{-4}$  and  $10^{-5}$  for RoBERTa and the positional transformer, respectively. The model was trained for 15 epochs, with early stopping if development set performance did not increase after five epochs.

Recall from the discussion in Section 2 that the input texts are pre-segmented into a list of “tokens” (hereafter *segments*) in the CURT dataset, but that these segments do not necessarily align with the tokens generated by the RoBERTa tokenizer. As such, for each input text  $t$ , we apply the tokenizer to each input segment  $t_i$  in  $t$  and concatenate the resulting tokens to pass as

Model	PT	DaBERTa	BERT+CRF	SpanBERT+CRF	RoBERTa+CRF
F1	0.8344	<b>0.8604</b>	0.8368	0.8390	0.8457

**Table 2**

Experimental results of the positional transformer (PT) and baseline architectures on the CURT dataset.

input to the model. For training, given a segment  $t_i$  with label  $l_i$ , each token within  $t_i$  receives the label  $l_i$ . For inference, we pool over the predicted labels for each token within the segment; we tested max-, min-, and mean-pooling, and found that mean-pooling yielded the highest F1 scores on the development set.

Finally, we augmented the data by POS-tagging each word in the input texts, using the NLTK *UnigramTagger*<sup>2</sup> trained on the Brown [18] corpus. While a more sophisticated POS-tagging model likely would have yielded better results, the pre-segmented nature of the input texts made applying a POS tagger with beyond-unigram complexity exceedingly difficult.

## 5. Results and Discussion

The positional transformer achieves an F1 score of **0.8344**, the highest-performing submission to CLAIMSCAN Task B, and the fifth-best out of the organizers’ baseline models (see Table 2). Unfortunately, due to issues with the submission portal, participants were unable to view their scores on after submission; this made optimizing performance with respect to the test set somewhat problematic. In particular, it was difficult to ascertain whether our model was “over-fitting” the development set, in the sense that we were not able to optimize our model/training hyperparameters with respect to the test set. We believe that the positional transformer could have achieved a higher F1 score on this task had that not been the case (as we observed F1 scores above 0.85 on the development set), and would have liked to perform ablation studies to this effect. Unfortunately, the test set labels for the task have yet to be released at this time.

## 6. Conclusion

In this paper, we introduced the positional transformer token classification architecture for claim span identification in the CLAIMSCAN (Task B) shared task. We found that this architecture outperforms all other submissions to this task, but falls short of the performance of some of the task organizers’ baseline models. Given certain limitations regarding the submission format of this task, however, we remain optimistic about the utility of the positional transformer for claim span identification. In the immediate future, we aim to conduct ablation studies on this model to identify the optimal hyperparameter configuration for this task. Additionally, we hope to evaluate the positional transformer on other sequence classification tasks, given the task-agnostic nature of this architecture.

<sup>2</sup><https://www.nltk.org/api/nltk.tag.html>



## References

- [1] M. Sundriyal, A. Kulkarni, V. Pulastya, M. S. Akhtar, T. Chakraborty, Empowering the fact-checkers! automatic identification of claim spans on Twitter, in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 7701–7715. URL: <https://aclanthology.org/2022.emnlp-main.525>. doi:10.18653/v1/2022.emnlp-main.525.
- [2] S. E. Toulmin, The uses of argument, Cambridge university press, 2003.
- [3] C. Stab, I. Gurevych, Parsing argumentation structures in persuasive essays, Computational Linguistics 43 (2017) 619–659.
- [4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).
- [5] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692 (2019).
- [6] Y. Tay, A. T. Luu, A. Zhang, S. Wang, S. C. Hui, Compositional de-attention networks, Advances in Neural Information Processing Systems 32 (2019).
- [7] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.
- [8] J. D. Lafferty, A. McCallum, F. C. N. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: ICML, 2001, pp. 282–289.
- [9] L. A. Ramshaw, M. P. Marcus, Text Chunking Using Transformation-Based Learning, Springer Netherlands, Dordrecht, 1999. URL: [https://doi.org/10.1007/978-94-017-2390-9\\_10](https://doi.org/10.1007/978-94-017-2390-9_10). doi:10.1007/978-94-017-2390-9\_10.
- [10] E. Cabrio, S. Villata, Five years of argument mining: A data-driven analysis., in: IJCAI, volume 18, 2018, pp. 5427–5433.
- [11] T. Chakrabarty, C. Hidey, S. Muresan, K. McKeown, A. Hwang, AMPERSAND: Argument mining for PERSuasive oNline discussions, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 2933–2943. URL: <https://aclanthology.org/D19-1291>. doi:10.18653/v1/D19-1291.
- [12] M. Taboada, W. C. Mann, Applications of rhetorical structure theory, Discourse studies 8 (2006) 567–588.
- [13] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. URL: <https://aclanthology.org/N19-1423>. doi:10.18653/v1/N19-1423.
- [14] L. Cheng, L. Bing, Q. Yu, W. Lu, L. Si, APE: Argument pair extraction from peer review and rebuttal via multi-task learning, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Lin-



guistics, Online, 2020, pp. 7000–7011. URL: <https://aclanthology.org/2020.emnlp-main.569>. doi:10.18653/v1/2020.emnlp-main.569.

- [15] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional lstm networks, in: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., volume 4, IEEE, 2005, pp. 2047–2052.
- [16] P. He, X. Liu, J. Gao, W. Chen, Deberta: Decoding-enhanced bert with disentangled attention, in: International Conference on Learning Representations, 2021.
- [17] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).
- [18] W. N. Francis, H. Kucera, Brown corpus manual, Letters to the Editor 5 (1979) 7.