Paragraph-based Transformer Pretraining for Multi-Sentence Inference

Anonymous ACL submission

Abstract

Inference tasks such as answer sentence selection (AS2) or fact verification are typically solved by fine-tuning transformer-based models as individual sentence-pair classifiers. Recent studies show that these tasks benefit from modeling dependencies across multiple candidate sentences 'jointly'. In this paper, we first show that popular pretrained transformers perform *poorly* when used for fine-tuning on multi-candidate inference tasks. We then propose a new pretraining objective that models the paragraph-level semantics across multiple input sentences. Our evaluation on three AS2, and one fact verification dataset demonstrates the superiority of our pretrained joint models over pretrained transformers for multi-candidate inference tasks.

1 Introduction

Pretrained transformers (Devlin et al., 2019; Liu et al., 2019; Clark et al., 2020) have become the de facto standard for several NLP applications, by means of fine-tuning on downstream data. There are several downstream NLP applications that require reasoning across multiple inputs candidates jointly towards prediction. Some popular examples include (i) Answer Sentence Selection (AS2) (Garg et al., 2020), which is a Question Answering (QA) task that requires selecting the best answer from a set of candidates for a question; and (ii) Fact Verification (Thorne et al., 2018), which reasons whether a claim is supported/refuted by multiple evidences. Inherently, these tasks can utilize information from multiple candidates (answers/evidences) to support the prediction of a particular candidate.

Pretrained transformers such as BERT are used for these tasks as cross-encoders by setting them as sentence-pair classification problems, i.e, aggregating inferences independently over each candidate. Recent studies (Zhang et al., 2021; Tymoshenko and Moschitti, 2021) have shown that these tasks benefit from encoding multiple candidates together, e.g., encoding five answer candidates per question in the transformer, so that the cross-attention can model dependencies between them. However, Zhang et al. only improved over the pairwise cross-encoder by aggregating multiple pairwise cross-encoders together (one for each candidate), and not by jointly encoding all candidates together in a single model.

In this paper, we first show that popular pretrained transformers such as RoBERTa perform *poorly* when used for jointly modeling inference tasks (e.g., AS2) using multi-candidates. We show that this is due to a shortcoming of their pretraining objectives, being unable to capture meaningful dependencies among multiple candidates for the fine-tuning task. To improve this aspect, we propose a new pretraining objective for 'joint' transformer models, which captures paragraph-level semantics across multiple input sentences. Specifically, given a target sentence s and multiple sentences (from the same/different paragraph/document), the model needs to recognize which sentences belong to the same paragraph as s in the document used.

Joint inference over multiple-candidates entails modeling interrelated information between multiple *short* sentences, possibly from different paragraphs or documents. This differs from related works (Beltagy et al., 2020; Zaheer et al., 2020) on transformers that model long contiguous inputs (documents) to get longer context for tasks such as machine reading and summarization.

We evaluate our pretrained multiple candidate-based joint models by (i) performing AS2 on ASNQ, WikiQA, TREC-QA datasets; and (ii) Fact Verification on the FEVER dataset. We show that our pretrained joint models substantially improve over the performance of transformers such as RoBERTa being used as joint models for multi-candidate inference tasks, as well as when being used as cross-encoders for sentence-pair formulations of these tasks.

2 Related Work

Multi-Sentence Inference: Inference over a set of multiple candidates has been studied in the past (Bian et al., 2017; Ai et al., 2018). The most relevant for AS2 are the works of Bonadiman and Moschitti (2020) and Zhang et al. (2021).

Transformer Pretraining Objectives: Masked Language Modeling (MLM) is a popular transformer pretraining objective (Devlin et al., 2019; Liu et al., 2019). Other models are trained using token-level (Clark et al., 2020; Joshi et al., 2020; Yang et al., 2020; Liello et al., 2021) and/or sentence-level (Devlin et al., 2019; Lan et al., 2020; Wang et al., 2019) objectives. REALM (Guu et al., 2020) uses a differentiable neural retriever over Wikipedia to improve MLM pretraining. Giorgi et al. (2021) propose a contrastive learning objective for cross-encoding two sentences coming from the same/different documents in a transformer.

Modeling Longer Sequences: Beltagy et al. (2020); Zaheer et al. (2020) reduce the asymptotic complexity of transformer attention to model very long inputs for longer context. DCS (Ginzburg et al., 2021) proposes a cross-encoder for the task of document-pair matching, while CDLM (Caciularu et al., 2021) specializes the Longformer for this same task and cross-document coreference resolution. These works encode a single contiguous long piece of text, which differs from our setting of having multiple *short* candidates, for a topic/query, possibly from different paragraphs and documents. Due to brevity of space, please refer to Appendix A for a detailed discussion of related work.

3 Multi-Sentence Transformers Models

3.1 Multi-sentence Inference Tasks

AS2: We denote the question by q, and the set of answer candidates by $C=\{c_1, \dots c_n\}$. The objective is to re-rank C and find the best answer A for q. AS2 is typically treated as a binary classification task: first, a model f is trained to predict the correctness/incorrectness of each c_i; then, the candidate with the highest likelihood of being correct is selected as an answer, *i.e.*, $A=\text{argmax}_{i=1}^{n} f(c_i)$. Intuitively, modeling interrelated information between multiple c_i's can help in selecting the best answer candidate (Zhang et al., 2021).

Fact Verification: We denote the claim by F, and the set of *evidences* by $C=\{c_1 \dots c_n\}$ that are retrieved using DocIR. The objective is to predict whether F is supported/refuted/neither using C (at least one evidence c_i is required for supporting/refuting F). Tymoshenko and Moschitti (2021) jointly model evidences for supporting/refuting a claim as they can complement each other.

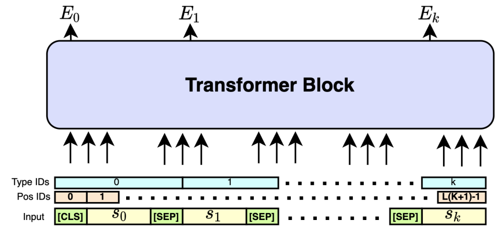

Figure 1: Multi-sentence 'Joint' transformer model. E_i refers to embedding for the question/each candidate.

3.2 Joint Model Architecture

For jointly modeling multi-sentence inference tasks, we use a monolithic transformer cross-encoder to encode multiple sentences using self-attention as shown in Fig 1. To perform joint inference over k sentences for question q or claim F, the model receives concatenated sentences $[s_0 \dots s_k]$ as input, where the first sentence is either the question or the claim ($s_0=q$ or $s_0=F$), and the remainder are k candidates $s_i=c_i$, $i=\{1 \dots k\}$. We pad (or truncate) each sentence s_i to the same fixed length L (total input length $L \times (k+1)$), and use the embedding for the [CLS] / [SEP] token in front of each sentence s_i as its embedding (denoted by E_i). Similar to Devlin et al., we create positional embeddings of tokens using integers 0 to $L(k+1)-1$, and extend the token type ids from $\{0, 1\}$ to $\{0 \dots k\}$ corresponding to $(k+1)$ input sentences.

3.3 Inference using Joint Transformer Model

We use the output embeddings $[E_0 \dots E_k]$ of sentences for performing prediction as following:

Predicting a single label: We use two separate classification heads to predict a single label for the input to the joint model $[s_0 \dots s_k]$: (i) **IE$_1$:** a linear layer on the output embedding E_0 of s_0 (similar to BERT) referred to as the Individual Evidence (IE$_1$) inference head, and (ii) **AE$_1$:** a linear layer on the average of the output embeddings $[E_0, E_1, \dots, E_k]$ to explicitly factor in information from all candidates, referred to as the Aggregated Evidence (AE$_1$) inference head. For Fact Verification, we use prediction heads IE$_1$ and AE$_1$.

Predicting Multiple Labels: We use two separate classification heads to predict k labels, one label each for every input $[s_1 \dots s_k]$ specific to s_0: (i) **IE$_k$:** a shared linear layer applied to the output embedding E_i of each candidate s_i , $i \in \{1 \dots k\}$ re-

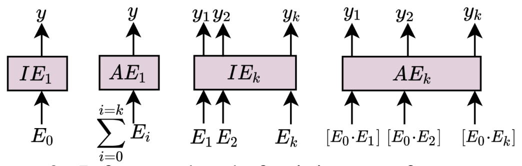

Figure 2: Inference heads for joint transformer model. E_i refers to embedding for the question/each candidate.

ferred to as k-candidate Individual Evidence (IE$_k$) inference head, and (ii) **AE$_k$:** a shared linear layer applied to the concatenation of output embedding E_0 of input s_0 and the output embedding E_i of each candidate s_i , $i \in \{1 \dots k\}$ referred to as k-candidate Aggregated Evidence (AE$_k$) inference head. For AS2, we use prediction heads IE$_k$ and AE$_k$. Prediction heads are illustrated in Figure 2.

3.4 Pretraining Paragraph-level Semantics

Long documents are typically organized into paragraphs to address the document's general topic from different viewpoints. The majority of transformer pretraining strategies have not exploited this rich source of information, which can possibly provide some weak supervision to the otherwise unsupervised pretraining phase. To enable joint transformer models to effectively capture dependencies across multiple sentences, we design a new pretraining task where the model is (i) provided with $(k+1)$ sentences $\{s_0 \dots s_k\}$, and (ii) tasked to predict which sentences from $\{s_1 \dots s_k\}$ belong to the same paragraph P as s_0 in the document D. We call this pretraining task Multi-Sentences in Paragraph Prediction (**MSPP**). We use the IE$_k$ and AE$_k$ prediction heads, defined above, on top of the joint model to make k predictions p_i corresponding to whether each sentence s_i, $i{\in}\{1 \dots k\}$ lies in the same paragraph $P \in D$ as s_0. More formally:

$$p_i = \begin{cases} 1 & \text{if } s_0, s_i \in P \text{ in } D \\ 0 & \text{otherwise} \end{cases} \qquad \forall i{=}\{1, \dots, k\}$$

We randomly sample a sentence from a paragraph P in a document D to be used as s_0, and then (i) randomly sample k_1 sentences (other than s_0) from P as positives, (ii) randomly sample k_2 sentences from paragraphs other than P in the same document D as hard negatives, and (iii) randomly sample k_3 sentences from documents other than D as easy negatives (note that $k_1{+}k_2{+}k_3{=}\ k$).

4 Experiments

We evaluate our joint transformers on three AS2 and one Fact Verification datasets. Note that we do not use GLUE (Wang et al., 2018) tasks as they only involve sentence pair classification.

4.1 Datasets

Pretraining: To eliminate any improvements stemming from usage of more data, we perform pretraining on the same corpora as RoBERTa: English Wikipedia, the BookCorpus, OpenWebText and CC-News. For our proposed pretraining, we randomly sample sentences from paragraphs as s_0, and choose $k_1{=}1, k_2{=}2, k_3{=}2$ as the specific values for creating positive and negative candidates for s_0. For complete details refer to Appendix B.

Fine-tuning: For AS2, we compare performance with MAP, MRR and Precision of top ranked answer (P@1). For fact verification, we measure Label Accuracy (LA). Brief description of datasets is presented below (details in Appendix B):

- **ASNQ:** A large AS2 dataset (Garg et al., 2020) derived from NQ (Kwiatkowski et al., 2019).
- **WikiQA:** An AS2 dataset (Yang et al., 2015) with questions from Bing search logs.
- **TREC-QA:** A popular AS2 dataset (Wang et al., 2007) containing factoid questions.
- **FEVER:** A popular dataset for fact extraction and verification (Thorne et al., 2018).

4.2 Experimental Details and Baselines

We use $k{=}5$ for our experiments (following Zhang et al., Tymoshenko and Moschitti), and perform continued pretraining starting from RoBERTa-Base using a combination of MLM and our MSPP pretraining for 100k steps with a batch size of 4,096. We use two different prediction heads, IE$_k$ and AE$_k$, for pretraining. For evaluation, we fine-tune all models on the downstream AS2 and FEVER datasets using the corresponding IE$_k$ and AE$_k$ prediction heads. We consider the pairwise RoBERTa-Base cross-encoder and RoBERTa-Base LM used as a joint model with IE$_k$ and AE$_k$ prediction heads as the baseline for AS2 tasks. For FEVER, we use several baselines: GEAR (Zhou et al., 2019), KGAT (Liu et al., 2020), Transformer-XH (Zhao et al., 2020), and three models from Tymoshenko and Moschitti: (i) Joint RoBERTa-Base with IE$_1$ prediction head, Pairwise RoBERTa-Base with max-pooling, and weighted-sum heads. For complete experimental details, refer to Appendix C.

4.3 Results

The results for AS2 tasks are presented in Table 1, averaged across 5 runs. From the table, we can see that the RoBERTa-Base when used as a joint model for multi-candidate inference using either

Model	ASNQ			WikiQA			TREC-QA		
	P@1	MAP	MRR	P@1	MAP	MRR	P@1	MAP	MRR
Pairwise RoBERTa-Base	61.8 (0.2)	66.9 (0.1)	73.1 (0.1)	77.1 (2.1)	85.3 (0.9)	86.5 (1.0)	87.9 (2.2)	89.3 (0.9)	93.1 (1.0)
Joint RoBERTa-Base → FT IE$_k$	3.4 (2.3)	8.0 (1.9)	10.0 (2.4)	19.7 (1.9)	39.4 (1.6)	40.3 (1.8)	30.9 (5.4)	41.9 (2.4)	50.8 (3.9)
Joint RoBERTa-Base → FT AE$_k$	3.6 (2.7)	8.0 (2.2)	10.2 (2.8)	18.7 (3.9)	39.0 (2.8)	39.7 (2.9)	29.7 (6.9)	42.3 (3.2)	49.2 (5.0)
(**Ours**) Joint MSPP IE$_k$ → FT IE$_k$	**63.0** (0.3)	67.2 (0.2)	**73.7** (0.2)	**82.7** (2.2)	**88.5** (1.5)	**89.0** (1.5)	**91.7** (2.2)	**91.1** (0.5)	**95.2** (1.3)
(**Ours**) Joint MSPP AE$_k$ → FT AE$_k$	**63.0** (0.3)	**67.3** (0.2)	**73.7** (0.2)	81.9 (2.6)	87.9 (1.4)	89.0 (1.5)	88.7 (0.8)	90.1 (1.0)	93.6 (0.6)

Table 1: Results(with std. dev. in parenthesis) on AS2 datasets with standard deviation. MSPP, FT refer to our pretraining task and fine-tuning respectively. We indicate the prediction head (IE$_k$/AE$_k$) used for both pretraining and fine-tuning. We underline statistically significant gains over the baseline (T-Test with 95% confidence level).

Model	ASNQ	WikiQA	TREC-QA
Pairwise RoBERTa-Base	61.8 (0.2)	77.1 (2.1)	87.9 (2.2)
Joint RoBERTa-Base → FT IE$_k$	25.2 (3.1)	24.6 (3.1)	57.6 (4.8)
Joint RoBERTa-Base → FT AE$_k$	25.4 (3.3)	26.4 (2.2)	60.9 (4.9)
(**Ours**) Joint MSPP IE$_k$ → FT IE$_k$	63.9 (0.8)	**82.7** (3.0)	**92.2** (0.8)
(**Ours**) Joint MSPP AE$_k$ → FT AE$_k$	**64.3** (1.1)	82.1 (1.1)	91.2 (2.9)

Table 2: P@1 of joint models for AS2 when re-ranking the answer candidates ranked in top-k by Pairwise RoBERTa-Base. Statistically significant results (T-Test 95%) are underlined. Complete results in Appendix D.

the IE$_k$/AE$_k$ prediction heads performs inferior to RoBERTa-Base used as a pairwise cross-encoder. Across 5 experimental runs, we observe that fine-tuning RoBERTa-Base as a joint model faces convergence issues (across various hyper-parameters) indicating that the MLM pretraining task is not sufficient to learn text semantics which can be exploited for multi-sentence inference.

Our MSPP pretrained joint models (with both IE$_k$, AE$_k$ heads) get significant improvements over the pairwise cross-encoder baseline and very large improvements over the RoBERTa-Base joint model. The former highlights modeling improvements stemming from joint inference over multiple-candidates, while the latter highlights improvements stemming from our MSPP pretraining strategy. Across all three AS2 datasets, our joint models are able to get the highest P@1 scores while also improving the MAP and MRR metrics.

To demonstrate that our joint models can effectively use information from multiple candidates towards prediction, we perform a study in Table 2 where the joint models are used to re-rank the top-k candidates ranked by the pairwise RoBERTa-Base cross-encoder. Our joint models can significantly improve the P@1 over the baseline for all datasets. The performance gap stems from questions for which the pairwise RoBERTa model was unable to rank the correct answer at the top position, but support from other candidates in the top-k helped the joint model rank it in the top position. Due to space limitations, refer to Appendix E for some qualitative examples highlighting this improvement.

The results for the FEVER task are presented in Table 3 and show that our joint models (pre-

Model	Dev	Test
GEAR	70.69	71.60
KGAT with RoBERTa-Base	78.29	74.07
Transformer-XH	78.05	72.39
Pairwise RoBERTa-Base + MaxPool	79.82	-
Pairwise RoBERTa-Base + WgtSum	80.01	-
Joint RoBERTa-Base + FT IE$_1$	79.25	73.56
(**Ours**) Joint Pre IE$_k$ + FT IE$_1$	**81.21** (0.24)	**74.39**
(**Ours**) Joint Pre IE$_k$ + FT AE$_1$	81.10 (0.15)	74.25
(**Ours**) Joint Pre AE$_k$ + FT IE$_1$	81.18 (0.14)	73.77
(**Ours**) Joint Pre AE$_k$ + FT AE$_1$	**81.21** (0.16)	74.13

Table 3: Results on FEVER dev and test sets. For our method, prediction heads (IE$_1$/AE$_1$) are only used for fine-tuning (FT), while for pretraining (Pre) we use the (IE$_k$/AE$_k$) heads. '-' denotes models not released publicly and have no reported results in any paper. Statistically significant results (T-Test 95%) are underlined.

trained with both the IE$_k$ and AE$_k$ heads and fine-tuned with the IE$_1$ and AE$_1$ heads) outperform all previous baselines considered, including the RoBERTa-Base joint model directly applied for multi-sentence inference.

Compute Overhead: A *simplified* latency analysis for AS2 (assuming sentence length L): pairwise cross-encoder will use k transformer steps with input length $2L$, while our joint model will use 1 step with input length $(k+1)\times L$. Since transformer attention scales quadratic on input length, our joint inference should take $\frac{(k+1)^2}{4k}$ times the inference time of the cross-encoder, which is 1.8 when k=5. However, when we fine-tune for WikiQA on one A100-GPU, we only observe latency increasing from $71s \rightarrow 81s$ (increase of only 14.1%). The input embeddings and feedforward layers vary linearly with input length, reducing overheads of self-attention. Refer to Appendix D.3 for details.

5 Conclusions

In this paper we have presented a multi-sentence cross-encoder for performing inference jointly on multiple sentences. We have proposed a novel pretraining task to capture paragraph-level semantics. Our experiments on 3 AS2 and 1 Fact verification datasets show that our pretrained joint models can outperform pairwise cross-encoders and pretrained LMs when directly used as joint models.

References

Qingyao Ai, Keping Bi, Jiafeng Guo, and W. Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 1987–1990, New York, NY, USA. Association for Computing Machinery.

Daniele Bonadiman and Alessandro Moschitti. 2020. A study on efficiency, accuracy and document structure for answer sentence selection.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew Peters, Arie Cattan, and Ido Dagan. 2021. CDLM: Cross-document language modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2648–2662, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Siddhant Garg and Alessandro Moschitti. 2021. Will this question be answered? question filtering via answer model distillation for efficient question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7329–7346, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7780–7788.

Dvir Ginzburg, Itzik Malkiel, Oren Barkan, Avi Caciularu, and Noam Koenigstein. 2021. Self-supervised document similarity ranking via contextualized language models and hierarchical inference. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3088–3098, Online. Association for Computational Linguistics.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895, Online. Association for Computational Linguistics.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations.

Luca Di Liello, Matteo Gabburo, and Alessandro Moschitti. 2021. Efficient pre-training objectives for transformers.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of ACL*.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189, Copenhagen, Denmark. Association for Computational Linguistics.

Luca Soldaini and Alessandro Moschitti. 2020. The cascade transformer: an application for efficient answer sentence selection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5697–5708, Online. Association for Computational Linguistics.

Dominik Stammbach and Elliott Ash. 2020. e-fever: Explanations and summaries forautomated fact checking. In *TTO*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Kateryna Tymoshenko and Alessandro Moschitti. 2021. Strong and light baseline models for fact-checking joint inference. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4824–4830, Online. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic. Association for Computational Linguistics.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Jiangnan Xia, Liwei Peng, and Luo Si. 2019. Structbert: Incorporating language structures into pre-training for deep language understanding.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Yi Yang, Scott Wen-tau Yih, and Chris Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. ACL - Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

Zeyu Zhang, Thuy Vu, and Alessandro Moschitti. 2021. Joint models for answer verification in question answering systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3252–3262, Online. Association for Computational Linguistics.

Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. 2020. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *The Eighth International Conference on Learning Representations (ICLR 2020)*.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180, Online. Association for Computational Linguistics.

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. GEAR: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901, Florence, Italy. Association for Computational Linguistics.

Appendix

A Related Work

Multi-Sentence Inference: Several research works (Bian et al., 2017; Ai et al., 2018) have explored performing inference over more than one, or the entire set candidates in the past. For the task of AS2, Bonadiman and Moschitti (2020) explore using using multiple answer candidates and improve the performance of older neural networks, however they fail to beat the performance of transformers. Zhang et al. (2021) use task-specific models (termed answer support classifiers) in addition to the AS2 transformer for performance improvements. For fact verification, Tymoshenko and Moschitti (2021) propose jointly embedding multiple evidence with the claim towards improving the performance of baseline pairwise cross-encoder transformers.

Transformer Pretraining: REALM (Guu et al., 2020) uses a differentiable neural retriever over Wikipedia to improve MLM pretraining. This differs from our pretraining setting as it uses additional data/knowledge stemming from the Wikipedia index to improve the pretrained LM for knowledge intensive tasks. DeCLUTR (Giorgi et al., 2021) uses a contrastive learning objective for cross-encoding two sentences coming from the same/different documents in a transformer. DeCLUTR is evaluated for sentence-pair classification tasks and embeds the two inputs independently without any cross-attention, which differs from our setting of embedding multiple candidates jointly for inference.

Modeling Longer Sequences: Longformer (Beltagy et al., 2020) uses a mixture of global and local attention to reduce complexity of full self-attention, to model very long contiguous inputs for longer context. For tasks with short sequence lengths, LongFormer works on par or slightly worse than RoBERTa (attributed to reduced attention computation). Big Bird (Zaheer et al., 2020) uses a hybrid attention (attending to random, nearby and the whole sequence) to model long contiguous inputs such as documents. Our proposed pretraining objective is complementary to these works, and can possibly be combined with them to enhance representations of longer inputs. DCS (Ginzburg et al., 2021) goes beyond the input length of the Longformer by means of a cross-encoder for the task of document-pair matching (for retrieval). DCS is related to our work as it uses a contrastive pretraining objective over two sentences extracted from the same paragraph, however different from our joint encoding of multiple sentences, DCS individually encodes the two sentences and then uses the InfoNCE loss over the embeddings. CDLM (Caciularu et al., 2021) specializes the Longformer to obtain document embeddings for the task of document-pair matching and cross-document coreference resolution. While the pretraining objective in CDLM exploits information from multiple documents, it differs from our setting of joint inference over multiple short sentences for tasks such as AS2 and fact verification.

B Datasets

We present the complete details for all the datasets used in this paper along with links to download them for reproducibility of results.

B.1 Pretraining Datasets

We use the Wikipedia[1], BookCorpus[2], OpenWebText (Gokaslan and Cohen, 2019) and CC-News[3] datasets for performing pretraining of our joint transformer models. We do not use the STORIES dataset as it is no longer available for research use [4]. After decompression and cleaning we obtained 6GB, 11GB, 38GB and 394GB of raw text respectively from the BookCorpus, Wikipedia, OpenWebText and CC-News.

B.2 Finetuning Datasets

We evaluate our joint transformers on three AS2 and one Fact Verification datasets. The latter differs from the former in not selecting the best candidate, but rather explicitly using all candidates to predict the label. Here are the details of the finetuning datasets that we use for our experiments along with data statistics for each dataset:

- **ASNQ:** A large-scale AS2 dataset (Garg et al., 2020)[5] where the candidate answers are from Wikipedia pages and the questions are from search

[1] https://dumps.wikimedia.org/enwiki/20211101/

[2] https://huggingface.co/datasets/bookcorpusopen

[3] https://commoncrawl.org/2016/10/news-dataset-available/

[4] https://github.com/tensorflow/models/tree/archive/research/lm_commonsense#1-download-data-files

[5] https://github.com/alexa/wqa_tanda

queries of the Google search engine. ASNQ is a modified version of the Natural Questions (NQ) (Kwiatkowski et al., 2019) dataset by converting it from a machine reading to an AS2 dataset. This is done by labelling sentences from the long answers which contain the short answer string as positive correct answer candidates and all other answer candidates as negatives. We use the dev. and test splits released by Soldaini and Moschitti[6].

Split	# Questions	# Candidates	Avg. # C/Q
Train	57,242	20,377,568	356.0
Dev	1,336	463,914	347.2
Test	1,336	466,148	348.9

Table 4: Data Statistics for ASNQ dataset

• **WikiQA:** An AS2 dataset released by Yang et al.[7] where the questions are derived from query logs of the Bing search engine, and the answer candidate are extracted from Wikipedia. This dataset has a subset of questions having no correct answers (all-) or having only correct answers (all+). We remove both the all- and all+ questions for our experiments ("clean" setting).

Split	# Questions	# Candidates	Avg. # C/Q
Train	2,118	20,360	9.6
Dev	122	1,126	9.2
Test	237	2,341	9.9

Table 5: Data Statistics for WikiQA dataset.

• **TREC-QA:** A popular AS2 dataset released by Wang et al.. For our experiments, we trained on the *train-all* split, which contains more noise but also more question-answer pairs. Regarding the dev. and test sets we removed the questions without answers, or those having only correct or only incorrect answer sentence candidates. This setting refers to the "clean" setting (Shen et al., 2017), which is a TREC-QA standard.

Split	# Questions	# Candidates	Avg. # C/Q
Train	1,226	53,417	43.6
Dev	69	1,343	19.5
Test	68	1,442	21.2

Table 6: Data Statistics for TREC-QA dataset.

• **FEVER:** A popular benchmark for fact extraction and verification released by Thorne et al. The aim is to retrieve evidences given a claim, and then identify whether the retrieved evidences support or refute the claim or if there is not enough information to make a choice. For supporting/refuting a claim, at least one of the retrieved evidences must support/retrieve the claim. Note that the performance on FEVER depends crucially on the retrieval system and the candidates retrieved. For our experiments, we are interested only in the fact verification sub-task and thus we exploit the evidences retrieved by Liu et al. using a BERT-based DocIR[8].

Split	# Claims	# Evidences	Avg. # E/C
Train	145,406	722,473	4.97
Dev	19,998	98,915	4.95
Test	19,998	98,839	4.94

Table 7: Statistics for the FEVER dataset where evidences has been retrieved using (Liu et al., 2020).

C Experimental Setup

C.1 Complete Experimental Details

Following standard practice, the token ids, positional ids and token type ids are embedded using separate embedding layers, and their sum is fed as the input to the transformer layers. We use k=5 for our experiments (following Zhang et al.; Tymoshenko and Moschitti), and perform continuous pretraining starting from the RoBERTa-Base checkpoint using a combination of MLM and our MSPP pretraining objective for 100,000 steps with a batch size of 4096. We use a triangular learning rate with 10,000 warmup steps and a peak value of $5 * 10^{-5}$. We use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. We apply a weight decay of 0.01 and gradient clipping when values are higher than 1.0. We set the dropout ratio to 0.1 and we use two different prediction heads for pretraining: IE_k and AE_k. We follow the strategy of (Devlin et al., 2019; Lan et al., 2020), and equally weight the the two pretraining loss objectives: MLM and MSPP.

For evaluation, we fine-tune all models on the downstream AS2 and FEVER datasets: using the same IE_k and AE_k prediction heads exploited in pretraining for AS2 and using either IE_1 or AE_1 prediction heads for FEVER. We finetune every model with the same maximum sequence length equal to $64 * (k + 1) = 384$ tokens. For ASNQ we train for up to 6 epochs with a batch size of 512 and a learning rate of 10^{-5} with the same Adam optimizer described above but warming up for only 5000 steps. We do early stopping on the MAP of the development set. For WikiQA and TREC-QA, we created batches of 32 examples and we used

[6] `https://github.com/alexa/wqa-cascade-transformers`

[7] `http://aka.ms/WikiQA`

[8] `https://github.com/thunlp/KernelGAT/tree/master/data`

a learning equal to $2 * 10^{-6}$ and 1000 warm up steps. We train for up to 40 epochs again with early stopping on the MAP of the development set. On FEVER, we use a batch size of 64, a learning rate of 10^{-5}, 1000 warm up steps and we do early stopping checking the Accuracy over the development set.

C.2 Baselines

For AS2, we consider two baselines: (i) pairwise RoBERTa-Base model when used as a cross-encoder for AS2, and (ii) RoBERTa-Base LM when used as a joint model with IE_k and AE_k prediction heads independently for AS2 tasks.

For FEVER, we use several recent baselines from Tymoshenko and Moschitti: (i) GEAR (Zhou et al., 2019), (ii) KGAT (Liu et al., 2020), (iii) Transformer-XH (Zhao et al., 2020), (iv) joint RoBERTa-Base with IE_1 prediction head (Tymoshenko and Moschitti, 2021), (v) pairwise RoBERTa-Base when used as a cross-encoder with max-pooling head (Tymoshenko and Moschitti, 2021), (vi) pairwise RoBERTa-Base when used as a cross-encoder with weighted-sum head (Tymoshenko and Moschitti, 2021).

C.3 Metrics

The performance of AS2 systems in practical applications is typically (Garg and Moschitti, 2021) measured using the Accuracy in providing correct answers for the questions (the percentage of correct responses provided by the system), also called the Precision-at-1 (P@1). In addition to P@1, we use Mean Average Precision (MAP) and Mean Reciprocal Recall (MRR) to evaluate the ranking produced of the set of candidates by the model.

For FEVER, we measure the performance using Label Accuracy (LA), a standard metric for this dataset, that measures the accuracy of predicting support/refute/neither for a claim using a set of evidences.

D Complete Results and Discussion

D.1 Results on AS2 with cascaded pairwise and Joint re-ranker

Below we present results of evaluating our joint models to re-rank the top-k candidates ranked by the pairwise RoBERTa-Base cross-encoder. Our joint models can significantly improve the P@1, MAP and MRR over the baseline for all datasets. The performance gap stems from questions for which the pairwise RoBERTa model was unable to rank the correct answer at the top position, but support from other candidates in the top-k helped the joint model rank it in the top position.

Model	**Dev**	**Test**
GEAR	70.69	71.60
KGAT with RoBERTa-Base	78.29	74.07
Transformer-XH	78.05	72.39
Pairwise BERT-Base	73.30	69.75
Pairwise RoBERTa-Base + MaxPool	79.82	-
Pairwise RoBERTa-Base + WgtSum	80.01	-
Joint BERT-Base	73.67	71.01
Joint RoBERTa-Base + FT IE_1	79.25	73.56
(**Ours**) Joint Pre IE_k + FT IE_1	<u>**81.21**</u> (0.24)	**74.39**
(**Ours**) Joint Pre IE_k + FT AE_1	<u>81.10</u> (0.15)	74.25
(**Ours**) Joint Pre AE_k + FT IE_1	<u>81.18</u> (0.14)	73.77
(**Ours**) Joint Pre AE_k + FT AE_1	<u>**81.21**</u> (0.16)	74.13
Methods with larger models and/or sophisticated retrieval		
DOMLIN++	77.48	76.60
DREAM	79.16	76.85

Table 8: Complete Results on FEVER dev and test sets. For our method, prediction heads (IE_1/AE_1) are only used for finetuning (FT), while for pretraining (Pre) we use the (IE_k/AE_k) heads. '-' denotes models that are not publicly released and have no reported results on the test split in their published paper. Statistically significant results (T-Test 95%) are underlined.

D.2 Results on FEVER

Here we present complete results on the FEVER dataset in Table 8, by also presenting some additional baselines such as: (i) pairwise BERT-Base cross-encoder (Tymoshenko and Moschitti, 2021), (ii) joint BERT-Base cross-encoder with IE_1 prediction head, (iii) DOMLIN++ (Stammbach and Ash, 2020) which uses additional DocIR components and data (MNLI (Williams et al., 2018)) for fine-tuning, (iv) DREAM (Zhong et al., 2020) that uses the XL-Net model. Note that comparing our joint models with (iii) and (iv) is unfair since they use additional retrieval components, datasets and larger models. We just include these results here for the sake for completeness. Interestingly, our joint models outperform DREAM and DOMLIN++ on the dev set without using additional retrieval and larger models.

D.3 Compute Overhead of Joint Models

Change in Number of Model Parameters: The transformer block of our joint inference model is identical to pretrained models such as RoBERTa, and contains the exact same number of parameters. Classification heads IE_1, IE_k and AE_1 all operate on the embedding of a single token, and are identical to the classification head of RoBERTa

Model	ASNQ			WikiQA			TREC-QA		
	P@1	MAP	MRR	P@1	MAP	MRR	P@1	MAP	MRR
Pairwise RoBERTa-Base	61.8	66.9	73.1	77.1	85.3	86.5	87.9	89.3	93.1
Joint RoBERTa-Base → FT IE$_k$	25.2	44.0	45.6	24.6	49.3	49.7	57.6	73.7	74.6
Joint RoBERTa-Base → FT AE$_k$	25.4	44.8	46.2	26.4	50.6	51.1	60.9	74.6	76.7
(**Ours**) Joint MSPP IE$_k$ → FT IE$_k$	63.9	71.3	73.1	**82.7**	**88.5**	**89.0**	**92.2**	**93.5**	**95.4**
(**Ours**) Joint MSPP AE$_k$ → FT AE$_k$	**64.3**	**71.5**	**73.4**	82.1	87.9	88.7	91.2	**93.5**	94.9

Table 9: Complete results of our joint models for AS2 datasets when **re-ranking the answer candidates ranked in top-k by Pairwise RoBERTa-Base**. MSPP, FT refer to our pretraining task and finetuning respectively. We indicate the prediction head (IE$_k$/AE$_k$) used for both pretraining and finetuning.

(AE_k operates on the concatenation of two token embeddings, and contains double the number of parameters as the RoBERTa). The maximum sequence length allowed for both the models is the same (512). The exact number of parameters of our joint model with AE_k and the RoBERTa model are $124,062,720$ and $124,055,040$ respectively.

Change in Inference Latency: While our joint model provides a longer input sequence to the transformer, it also reduces the number of forward passes that need to be done by a pairwise cross-encoder. A *simplified* latency analysis for AS2 (assuming each sentence has a length L): pairwise cross-encoder will need to make k forward passes of the transformer with a sequence of length $2L$ (q with each candidate c_i), while our joint model will only need to make 1 forward pass of the transformer with input length $(k+1)\times L$ (q with k candidates). Transformer self-attention is quadratic in input sequence length, so this should lead to the inference time of out joint model being $\frac{(k+1)^2}{4k}$ times the inference time of the cross-encoder. However, the input embedding layer and the feedforward layers are linear in input sequence length, so this should lead to a reduction in the inference time of our joint model by $\frac{(k+1)}{2k}$ times the inference time of the cross-encoder. Empirically, when we fine-tune for WikiQA on one A100-GPU, we only observe latency increasing from $71s \rightarrow 81s$ (increase of only 14.1%).

E Qualitative Examples from AS2

We present some qualitative examples from the three AS2 datasets highlighting cases where the pairwise RoBERTa-Base model is unable to rank the correct answer on the top position, but our pretrained joint model (Joint MSPP IE$_k$ → FT IE$_k$) can do this using supporting information from other candidates in Table 10.

ASNQ
Q: Who invented the submarine during the civil war? **A1:** H.L. Hunley , often referred to as Hunley , was a submarine of the Confedera **A2:** Hunley , McClintock , and Baxter Watson first built Pioneer , which was tested in February 1862 in the Mississippi River and was later towed to Lake Pontchartrain for additional trials . **A3:** She was named for her inventor, Horace Lawson Hunley , shortly after she was taken into government service under the control of the Confederate States Army at Charleston , South Carolina. **A4:** 1864 painting of H.L. Hunley by Conrad Wise Chapman History Confederate States Name : H.L. Hunley Namesake : Horace Lawson Hunley Builder : James McClintock Laid down : Early 1863 Launched : July 1863 Acquired : August 1863 In service : February 17 , 1864 Out of service : February 17, 1864 Status : Awaiting conservation General characteristics Displacement : 7.5 short tons (6.8 metric tons) Length : 39.5 ft **A5:** Johan F. Carlsen was born in Ærøskøbing April 9, 1841.
WikiQA
Q: What is the erb/heart? **A1:** Heart valves are labeled with "B", "T", "A", and "P".First heart sound: caused by atrioventricular valves - Bicuspid/Mitral (B) and Tricuspid (T). **A2:** Second heart sound caused by semilunar valves – Aortic (A) and Pulmonary/Pulmonic (P). **A3:** Front of thorax , showing surface relations of bones , lungs (purple), pleura (blue), and heart (red outline). **A4:** In cardiology, Erb's point refers to the third intercostal space on the left sternal border where sS2 is best auscultated . **A5:** It is essentially the same location as what is referred to with left lower sternal border (LLSB).
TREC-QA
Q: When was the Khmer Rouge removed from power ? **A1:** Sihanouk was named head of state after the Khmer Rouge seized power in 1975, but was locked in his palace by the communists as they embarked on their brutal attempt to create an agrarian utopia . **A2:** When a Vietnamese invasion drove the Khmer Rouge from power in 1979 , Duch fled with other Khmer Rouge leaders into the jungles. **A3:** Religious practices were revived after the Khmer Rouge were driven from power by a Vietnamese invasion in 1979 **A4:** Moreover , 20 years after the Khmers Rouges were ousted from power , Cambodia still struggles on the brink of chaos , ruled by the gun , not by law . **A5:** Sihanouk resigned in 1976 , but the Khmer Rouge kept him under house arrest until they were driven from power by an invading Vietnamese army in 1979 .

Table 10: Qualitative examples from AS2 datasets where the pairwise RoBERTa-Base model is unable to rank a correct answer for the question at the top position, but our joint model (Joint MSPP $\text{IE}_k \rightarrow$ FT IE_k) can. Here we present the answers $A1, \ldots, A5$ in their ranked order by the pairwise RoBERTa-Base model. For all these examples we highlight the top ranked answer by the pairwise RoBERTa-Base model in red since it is incorrect.