EECS 595 Project Final Report Team 14

Jake Olkin, Rui Guo, Anthony Liang
University of Michigan
{jolkin,guorui,aliangdw}@umich.edu

Abstract

In the NLP community, many benchmark datasets and tasks have been created to enhance machines abilities to understand, interpret, and communicate with human language. For this project, we designed models to benchmark on three different tasks: CommonsenseQA, Conversation Entailment, and Everyday Actions in Text (EAT). Our model achieved 83.29% accuracy on validation set of CommonsenseQA task. We achieved 83.46% accuracy on conversational entailment task. For EAT task, we achieved 86.59% accuracy and 80.16% F1 score on the plausibility and breakpoint prediction. In this report, we present the models we designed, and also ablation studies on the design choices of our models.

1 Introduction

Recently, NLP research is increasingly focused on applying new deep learning methods to perform a wide-range of language related tasks from machine translation to dialogue systems. In this project, we benchmarked the performance of three models that we designed on three separate tasks: CommonsenseQA, Conversation Entailment, and Everyday Actions in Text. Each of these tasks, and associated data sets, presented unique challenges that required three different approaches to prediction. However, all of the models we designed were centered around using a pretrained transformer, so any innovations took the form of preprocessing techniques, or refactoring the problem to work better with a transformer.

The CommonsenseQA task (Talmor et al., 2018) is centered around answering questions. The model is given a natural language question, and five multiple choice answers to select from. Unlike other QA tasks, the CommonsenseQA tasks requires a model to have commonsense knowledge that might be trivial to humans but might be hard to learn by models. This was the largest dataset we were given to work with, containing 12,247 samples.

The Conversation Entailment task (Zhang and Chai, 2009) requires the model to infer information from dialogue. Given a short natural language exchange between two speakers, the model must identify whether or not a hypothesis sentence can be inferred from the conversation. This dataset was significantly smaller than the CommonsenseQA dataset, containing only 520 samples.

Lastly, the Everyday Actions in Text dataset requires a model to identify the plausibility of specific actions given a certain context. The model is given a story where each sentence describes an action taken. It must both determine if the story is plausible overall, and if it is not plausible, it must determine which sentence is implausible in context of their preceding sentences. This is also a small dataset, with only 1044 samples, and is the only dataset from the group that has two separate subtasks associated with it.

In this project, we designed our models for these three benchmark datasets. We based our models on transformer based contextual embedding models and designed task-specific structures for each task. To summarize, our main work includes:

(1) Designing and training customized models on different benchmark datasets

(2) Applying different preprocessing methods to all tasks and investigating their effects

(3) Investigating choices of model structures in ablation studies

2 Related Work

Benchmark datasets Recently, many benchmark datasets have been created by the NLP community to measure the performance of models. These benchmarks are mainly designed to measure models' linguistic knowledge, common knowledge and commonsense knowledge (Storks et al., 2019). Famous benchmarks include Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2018), Recognizing Textual Entailment (RTE) Challenge (Dagan et al., 2005), Multi-Genre Natural Language Inference (MNLI) (Williams et al., 2018), Situations with Adversarial Generations (SWAG) (Zellers et al., 2018). These benchmarks have different task forms, such as question answering, textual entailment, plausibility inference. In our project, we are focusing on all these three kinds of benchmarks.

Contextual Embedding Recently, contextual embedding approach has outperformed the classic word embedding approach (Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014)) on many benchmarks. People use LSTM based approaches, such as ELMo (Peters et al., 2018), or transformer-based approaches, such as BERT (Devlin et al., 2018), GPT (Radford, 2018) to extract word embedding based on the context in which they appear. These contextual embedding models can be easily adapt to downstream tasks by building task-specific structures on top of the contextual embedding and fine-tuning with task-specific data. In our project, we design task-specific structures for three benchmarks and evaluate the performance of different design choices.

3 Computational Models

3.1 Commonsense QA

For the CommonSense QA task, we are given as input a natural language question and five corresponding answer choices. The model is tasked with predicting the correct answer choice. In our method, we first format the input into a list of delimiter-separated sequences for each question and answer choice pair:

$$[CLS]\ \textbf{Question}\ [SEP]\ \textbf{A1}\ [SEP]$$
$$[CLS]\ \textbf{Question}\ [SEP]\ \textbf{A2}\ [SEP]$$
$$[CLS]\ \textbf{Question}\ [SEP]\ \textbf{A3}\ [SEP]$$
$$[CLS]\ \textbf{Question}\ [SEP]\ \textbf{A4}\ [SEP]$$
$$[CLS]\ \textbf{Question}\ [SEP]\ \textbf{A5}\ [SEP]$$

where $A1 - A5$ represent each answer choice. We use a pretrained tokenizer to encode the input sequence into tokens. We also pad the input sequence to a maximum length of 64. We compute attention masks to avoid performing attention on the padded tokens. Finally, we compute a one hot vector for separating the token indices between the question and the answer choices. We use these several pieces of information to create a feature vector for each QA example. These examples are then batched and fed into the transformer model during training. Mathematically, this process can be represented as:

$$z_{i,j} = Transformer([q_i, c_{ij}], a_i, p_i) \quad (1)$$

where $z_{i,j}$ represent the output logit for ith question, jth answer pair, q_i is the question token, c_{ij} is the jth answer choice, a_i is the attention mask, and p_i is the token type vector.

We formulate the Question Answering task as multi-class classification. After retrieving the logit outputs from the transformer model, we apply a softmax to convert the logits into a probability distribution. We then take an argmax over all the answers choices and select the one that yields the highest probability score. During training, we use cross entropy loss.

$$prediction_i = \arg\max_j(Softmax(z_{i,j})) \quad (2)$$

A visual illustration of our method is shown in Figure 1.

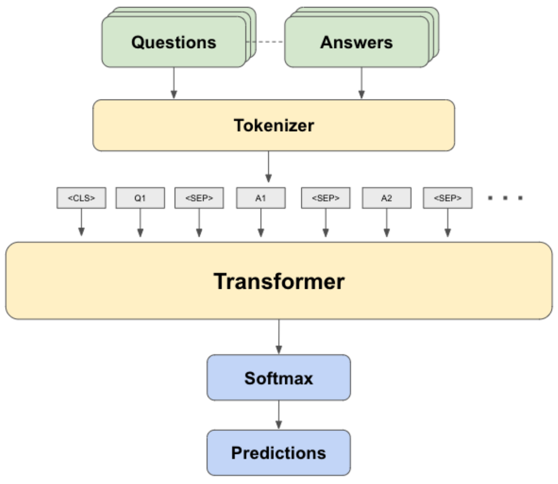

Figure 1: Block diagram of our architecture for the CommonsenseQA task

3.2 Conversational Entailment

For the Conversational Entailment task, we are given a dialog D from two speakers and a hypothesis H. Our aim is to infer whether the hypothesis can be entailed from the dialog.

Similar with the Commonsense QA task, we will build a binary classifier on the contextual embedding of the dialog and hypotheses, generated from a transformer-based encoder. We format the input as a delimiter-separated sequence of dialog and hypothesis.

$$[CLS]\ \textbf{Dialog}\ [SEP]\ \textbf{Hypothesis}\ [SEP] \tag{3}$$

The concatenated sequence will be tokenized and pass through a transformer(such as Bert(Devlin et al., 2018), RoBERTa(Liu et al., 2019)). Then we apply a binary classifier on the contextual embedding of $[CLS]$ token and use cross entropy loss to train the network. In our experiments, we use RoBERTa pretrained on MNLI dataset(Williams et al., 2018) as our base model.

Since our premise takes the form of dialog and it may contain more unstructured syntax than other kinds of more formal corpora, such as news letters or textbooks. For example, there may be slips of tongue during the conversation as well as heavy use of colloquial words(Um, Huh, I mean...). On the other hand, if the concatenated dialog loses speaker information, the model cannot understand the conversational process from the concatenated text, since there is no sign of changes of speakers.

Therefore, to make the dialog easier to understand by transformers, we applied several preprocessing techniques:

(a) We removed repeated words in the dialog. These words are labelled in the action tag file. We keep the second appearances of words, which are more confident than the first ones. (e.g *[I, + I] [use my, + only use my] credit card for ...* will be processed to *I only use my credit card for ...*)

(b) We add speakers to each turn in the dialog. This is aimed to indicate the speaker, since the hypotheses are sometimes only associated with one of two speakers. (e.g *SpeakerA: Are you on a regular exercise program right now? SpeakerB: Yes.*)

(c) We aligned the order of speakers. We noticed that in some conversations, *A* speakers first while in others, *B* speakers first. To remove this discrepancy, we exchange *speakerA* and *speakerB* in the hypothesis if *B* speaks first in the conversation.

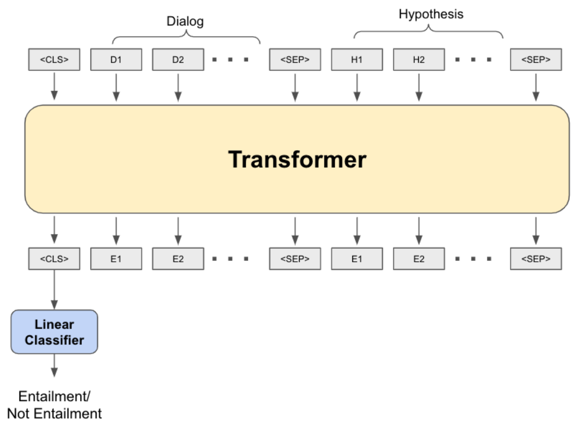

Figure 2: Block diagram of our architecture for the Conversational Entailment task

3.3 Everyday Actions in Text

For the Everyday Actions in Text task, we are given a story S which consists of sentences $S_1, S_2 ... S_n$ describing the actions taken by an actor. Our goal consists of two subtasks. The first subtask is to determine if the story as a whole is plausible or not. The second subtask is to determine which sentence causes the story not to be plausible overall (also called the breakpoint of the story).

To solve this task, we will train a model for subtask 2, which implicitly solves subtask 1. To solve the problem of breakpoint identification, we intend to reformulate the problem as a binary classification problem on the embedding of each individual sentence in context of the greater story, generated from a transformer-based contextual encoder.

Much like the CommonsenseQA and Conversation Entailment tasks, we format the input into a sequence of tokens for each story:

$$[CLS]\mathbf{S_1}\ \ \mathbf{S_2}\ ...\ \mathbf{S_n}[SEP] \tag{4}$$

where S_i is a sequence of tokens encoded from ith sentence. We also store along with this formatted inputs, a list of indices that represents the position of each sentence in the story, which is used during breakpoint prediction.

Using these tokenized sequences, we process our input through a transformer to produce an embedding of all of the sentences. We then separate this long embedding into each of the different sentences based on the spans we produced earlier. For each story, we get a list of sequences of embeddings e_i, which correspond to the ith sentence in that story. Each sequence in the list can have varying length N_i which depends on the length of each sentence.

We use $e_{i,j}$ to denote the embedding of jth token in ith sentence.

From the individual sentence embedding e_i, we then extract a single feature vector z_i representing of the entire sentence. This was done either by taking the first value (Equation 5), the maximum value (Equation 6) or the average value (Equation 7). We compare these methods in Section 5.3.

$$z_i = e_{i,0} \tag{5}$$

$$z_i = \max_j(e_{i,j}) \tag{6}$$

$$z_i = \frac{1}{N_i} \sum_{j=0}^{j=N_i-1} e_{i,j} \tag{7}$$

This feature z_i, after passing through a linear classifier, represents the probability that sentence it corresponds to is the breakpoint of the story, $P(y|s_i, S)$. This probability is dependent on both the story and sentence since we treat the all sentences from the same story as one entity in the transformer and we extract embedding according to indices of each sentence.

During inference, we examine each sentence to determine if it is a breakpoint. If there is not sufficiently high probability that any individual sentence is a breakpoint, then we deem that the story is plausible as a whole. Otherwise, we select the sentence with highest probability as the breakpoint.

Since the task revolves around the actions taken by a single actor, we realized after examining the training data that the name of the actor is irrelevant to determining the plausibility of the story. Therefore, to simplify the stories, we changed the name of the actor in every story to always be ”Tom”, so the model would not have to learn to deal with different names.

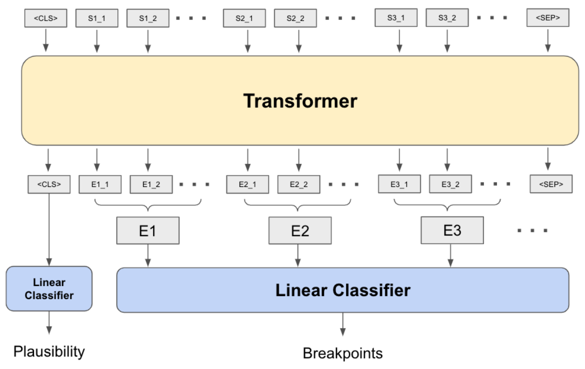

Figure 3: Block diagram of our architecture for the Everyday Actions in Text task

4 Experimental Results

4.1 CommonsenseQA

Evaluation We split our dataset into train, test and validation sets. There are 9741 training examples and 1221 validation examples. We evaluate two different transformer models: *bert-base-uncased* and *roberta-large*. We also experimented with using the *electra-large-discriminator* model (Clark et al., 2020) to pre-train BERT. Using the ground-truth answer choices provided in the dataset, we measure and report the validation accuracy scores from applying these two models.

Table 1: Model accuracy for CommonsenseQA Task

Model	Validation Accuracy
bert-base-uncased	0.5864
roberta-large	0.7436
electra-large-discriminator	**0.8329**

Implementation details For training our CSQA models, we use the Adam optimizer with a base learning rate of 2e-5. Because of the large size of the training data, we used a small batch size of 16. We use the first epoch as warmup. We train our model for a total of 5 epochs and measure the validation accuracy after every epoch of training. We limit the maximum length of the input sequence to 64 tokens. We use the Huggingface implementation (Wolf et al., 2019) of both BERT and RoBERTa models as well as the pre-trained Electra model.

Results Our results are listed in Table 1. Notably, the results that we obtain are similar to the ones reported on the CommonSenseQA leaderboard.

4.2 Conversational Entailment

Evaluation We evaluate different settings of our model on the conversational entailment data on the validation set. Since there are only 520 labelled examples available, we use 5-fold cross validation to make the evaluation metrics more stable. We use stratified fold split and make sure number of positive/negative samples and number of different types of hypothesis are kept similar in all fold splits. Each model is trained 5 times on different data split and we report the average accuracy on these 5 folds. We use 0.5 as the threshold.

Implementing Details We use RoBERTa(Liu et al., 2019) pretrained on MNLI(Williams et al., 2018)(from Huggingface (Wolf et al., 2020)) as our base model. We apply a dropout layer with probability 0.5 before the last classifier. We use Adam as the optimizer and our base learning rate is 2e-5. We do not use weight decay. We have 3 epochs for warmup and another 12 epochs for training with a cosine LR scheduler. We apply a layer LR-decay of 0.9 (e.g LR in last layer is the base LR, and will decay by 0.9 after each layer). We use 10 as our batchsize and we enable mixed-precision training from Pytorch. We measure the validation accuracy after every epoch of training and pick the epoch with highest validation accuracy.

Results Our best result is reported in the last row of Table 3. Our best model can reach 83.4% on average over all 5 folds. In our submission to the competition, we submit the average of predictions from models trained on 5 folds.

Table 2: Ablation study on choice of transformers in Conversational Entailment, all models are trained with the same settings and prepossessing, except the layer LR decay for Bert-base-cased is 0.85

Backbone	Accuracy
Bert-base-cased	0.623
RoBERTa-large	0.670
Electra-large-discriminator	0.709
RoBERTa-large-mnli	**0.834**

4.3 Everyday Actions in Text

Evaluation We evaluated different configurations of our EAT model against a validation set which we separated from the rest of our data prior to each training run. Since this is a smaller data set with only 1044 data points, we used 5-fold cross validation to create a more stable metric. During each fold, we split our data to ensure that the distribution of labels in our validation set approximated the distribution of those labels in our training set. We report the average metrics over each of these 5 folds.

Unfortunately, due to bias in the data, some labels were very sparsely represented in the validation set due to their low overall presence in the complete data set, as visualized in Figure 4. This lack of certain labels caused additional instability in our evaluation metrics.

Implementation Details We use the Electra model (Clark et al., 2020) as our base. We then extract feature embedding from the Electra model output by taking the first token from each sentence. Then, we apply a dropout layer of probability 0.5 to the extracted features before the applying the final linear classifier. We use Adam as our optimizer and a base learning rate of 1e-5. We use no weight decay. We use 3 training epochs as warm up, and then another 12 epochs for training with a cosine LR scheduler. We use a batch size of 16. After each epoch we evaluate our model on the validation set, and then use the metrics from the most accurate model in our 5-fold average.

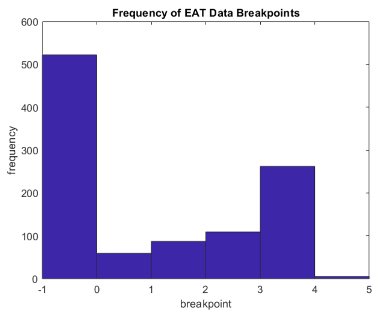

Figure 4: Histogram of data points in the EAT data set grouped by breakpoint.

Results Our best result is shown in the first line in Table 4. The model can achieve 86.59% accuracy on plausibility prediction and 80.16% F1 on breakpoint prediction. In our submission to the competition, we submitted a hard vote of predictions from 5 fold models.

5 Discussion

5.1 CommonsenseQA

For the CommonsenseQA task, we experimented with different pretrained backbones and transformer architectures. We achieve the best results using the pretrained Electra large discriminator model. If time permitted, we would like to incorporate relations from ConceptNet to improve the model performance. Given the nature of the QA dataset that we are predicting on, it makes sense that we leverage some mechanism of injecting common sense knowledge into the model. Many of the questions in the dataset require that the model predict answers that are out of context. For example, one of the questions is *"Where would I not want a fox"* with the answer choices being *"hen house, england, mountains, english hunt, california"*. Answering this question requires background knowledge of concepts that the model might not have been exposed to in the training dataset. A simple and likely effective solution following (Xu et al., 2020) would be to extract concepts from the ques-

Table 3: Ablation study on effect of preprocessing techniques in Conversational Entailment. The reported numbers are validation accuracy averaged on 5 folds. (hypotheses of *n/a* type are added to *desire* group)

Prepossessing	Belief	Fact	Desire	Intent	Overall Accuracy
None	0.775	0.759	0.769	0.852	0.775
+Speaker order alignment	0.763	0.771	0.794	**0.888**	0.787
+Speaker indicator	0.825	0.823	**0.846**	0.833	0.827
+Remove repeats (The best model)	**0.837**	**0.847**	0.769	0.814	**0.834**

Table 4: Ablation Study on effect of different feature extraction and preprocessing techniques in Everyday Actions in Text. The reported numbers are the average of the listed metrics evaluated over 5 folds.

Extraction Method	Actor Modification	Accuracy	F1-Measure	Precision	Recall
First	Yes	**0.8659**	0.8016	**0.8555**	0.7837
Max	Yes	0.8524	0.7996	0.8328	0.7749
Average	Yes	0.8611	**0.8047**	0.8228	0.7888
First	No	0.8563	0.7925	0.7835	**0.8318**
Max	No	0.8611	0.7778	0.8072	0.7558
Average	No	0.8390	0.75634	0.7809	0.7336

tion and answer choices and find a related triplet between these concepts. This triple is then feed as additional input to the language model and they show that this simple augmentation can achieve 80.7% accuracy on the official CommonsenseQA leaderboard.

5.2 Conversational Entailment

For the conversational Entailment task, we do ablation study to understand the effect of our preprocessing methods and also investigate the performance of different transformer backbones.

Firstly, we conduct experiments on our preprocessing methods. We use the same training settings and same evaluation metrics with our best model. Starting from a model with no preprocessing in the dialog, in each time, we add one preprocessing method to our model. We report the accuracy on cross validation sets and also the accuracy of 4 different kinds of hypothesis in Table 3. Note that hypotheses of *desire* and *intent* types only account for less than 20% data.

As expected, using preprocessing to make the dialog more formatted can significantly improve the overall accuracy and the speaker indicator improves the most. Interestingly, these preprocessing techniques improves *belief* and *fact* hypotheses a lot while hurt the performance of other types. We think this may because *belief* and *fact* hypotheses are easier to infer by checking the meaning of hypothesis and dialog. We give indications on where to find these information in the dialog by adding speaker indicator and align the speaker order. We also remove the distraction content in the dialog so that the searching process will become easier. However, for *desire* and *intent* type, repeat of words sometimes can indicate the speaker's feeling. So removing repeats may hurt the performance on these two types.

Secondly, we also analyzed the choice of transformers. We used same training settings as our best model, except that we change the layer LR decay to 0.85 for *Bert-base-cased*, since it has fewer layers. Our results are shown in Table 2. We find that using larger transformers may improve the performance and using Multi-NLI(Williams et al., 2018) pretrained transformers can improve more than 16% in accuracy. This is because the aim of MNLI dataset is to determine the entailment relationship, which is the same with our task and contains over 392k training samples. So knowledge about entailment learned from MNLI can be transferred to conversational entailment and improve the results.

5.3 Everyday Actions in Text

In the ablation study, we investigate the effect of using three different feature extraction techniques, as well as the application of a preprocessing method. Our feature extraction methods are

(a) "First": we use the value from the first token from each sentence embedding

(b) "Max": We use the largest value from each sentence embedding

(c) ”Average”: We use the average value over the entire sentence embedding

For our preprocessing method, we investigated the effects of modifying the actor (which is always the first token of each sentence) to always use the same name. This would create a more uniform form to each sentence, since the model would not have to account for different names, which hold no influence over the plausibility of the story in this data set.

We found that, we had the best accuracy, and precision when using first token extraction and changing the name of the actor in each sentence to be consistent. These settings also produced F1-Measure and recall scores comparable to the best results from other models. We believe this setting performed well compared to the other entries because, the majority of the plausibility information in the Electra model is stored in the token representing the actor in each story, which also is the first token in any given sentence. In general, the unified name change resulted in an increase in all metrics given a specific extraction method, excluding recall for the ”first” extraction method, however we believe this is due to how macro-recall can be an unstable metric due to the volatility of correctly predicting the small number of data points with a breakpoint of 5.

6 Conclusion

Throughout this project, we designed models on three different benchmark datasets. Our model achieved 83.29% accuracy on validation set of CommonSenseQA task. We achieved 83.46% accuracy on conversational entailment task. For EAT task, We achieved 86.59% accuracy and 80.16% F1 score on the plausibility and breakpoint prediction. We found that proprocessing techniques that simplified the data would increase the performance of our models, because it reduced the variance in data points that the model would have to learn to accommodate. Additionally, we found that choosing the right pre-trained backbone for our transformer also had a drastic effect on our models' abilities to perform each task.

References

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

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, page 177–190, Berlin, Heidelberg. Springer-Verlag.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26, pages 3111–3119. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

A. Radford. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822.

Shane Storks, Qiaozi Gao, and Joyce Y. Chai. 2019. Commonsense reasoning for natural language understanding: A survey of benchmarks, resources, and approaches. *CoRR*, abs/1904.01172.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *CoRR*, abs/1811.00937.

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. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. 2020. Fusing context into knowledge graph for commonsense reasoning.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. *CoRR*, abs/1808.05326.

Chen Zhang and Joyce Chai. 2009. What do we know about conversation participants: Experiments on conversation entailment. In *Proceedings of the SIGDIAL 2009 Conference*, pages 206–215, London, UK. Association for Computational Linguistics.