Dartmouth at SemEval-2022 Task 6: Detection of Sarcasm

Rishik Lad, Weicheng Ma, Soroush Vosoughi
Department of Computer Science, Dartmouth College
{rishik.lad.23,weicheng.ma.gr,soroush.vosoughi}@dartmouth.edu

Abstract

This paper introduces the result of Team Dartmouth's experiments on each of the five subtasks for the detection of sarcasm in English and Arabic tweets. This detection was framed as a classification problem, and our contributions are threefold: we developed an English binary classifier system with RoBERTa$_{\text{BASE}}$, an Arabic binary classifier with XLM-RoBERTa$_{\text{BASE}}$, and an English multilabel classifier with BERT$_{\text{BASE}}$. Preprocessing steps are taken with labeled input data prior to tokenization, such as extracting and appending verbs/adjectives or representative/significant keywords to the end of an input tweet to help the models better understand and generalize sarcasm detection. We also discuss the results of simple data augmentation techniques to improve the quality of the given training dataset as well as an alternative approach to the question of multilabel sequence classification. Ultimately, our systems place us in the top 14 participants for each of the five subtasks.

1 Introduction

Sarcasm is a form of irony that occurs when there is a discrepancy between the literal and intended meanings of a text or utterance. This discrepancy typically manifests itself in the form of dislike, contempt, or derogation. Furthermore, sarcasm can be divided into multiple types: general sarcasm, irony, satire, understatement, overstatement, and rhetorical question.

This task concerns itself with the detection of sarcasm in online tweets (Abu Farha et al., 2022). This is an important issue to solve because the nature of sarcasm can interfere with the effectiveness of natural language processing models, particularly when conducting sentiment analysis, opinion mining, or other emotion-based tasks. Machine learning models deployed for such business use cases can be negatively impacted when processing sarcastic texts and provide inaccurate results, thereby harming an organization's bottom line. Therefore, it is critical that machine learning models be developed that can understand how to detect, ingest, and truly understand sarcasm.

This paper discusses how to identify sarcastic tweets in binary and multi-label classification contexts for both English and Arabic, as directed by SemEval-2022 Task 6 (Abu Farha et al., 2022). There are five subtasks: general sarcasm detection in standalone English and Arabic tweets (Task A), identification of sarcastic category in an English tweet (Task B), and identification of the sarcastic tweet in an English/Arabic pair of tweets (Task C).

Our experiments show that pre-trained transformer models demonstrate a strong ability of solving most of the subtasks of this challenge. Specifically, we fine-tune a RoBERTa$_{\text{BASE}}$ model to detect the presence of sarcasm in tweets for the English components of Tasks A and C. For the Arabic version of Tasks A and C, we apply an XLM-RoBERTa$_{\text{BASE}}$ model. For Task B, which is framed as a multilabel sequence classification problem, after experimenting with several RoBERTa and BERT models, we report the performance of a BERT$_{\text{BASE}}$ model which is fine-tuned to detect the categories of sarcasm in English tweets, if any.

While these models perform reasonably well in the evaluations, the imbalanced distributions of labels and poor annotation quality for some instances introduce unexpected noise to the fine-tuning process of these models and harm their performance in the evaluations. Despite our effort to augment unrepresented classes in the training data, simple data augmentation approaches do not show clear positive effects on the models' performance. More advanced data augmentation methods could be tried to trigger notable performance improvements on the challenge test dataset.

Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022), pages 912 - 918

2 Approaches

As mentioned above, there are three systems designed for the five subtasks. We discuss the model architecture, input processing, and other key elements for each of the systems below.

2.1 RoBERTa for Binary Classification

The English component for Tasks A and C require us to distinguish whether a tweet is sarcastic, either as a standalone text (Task A) or in comparison to another tweet (Task C). Both of these tasks were framed as binary classification problems and our solution leveraged the RoBERTa model (Figure 1) from Facebook AI to achieve this by producing rich feature representations from the inputs.

In particular, RoBERTa builds upon the Bidirectional Encoder Representations from Transformers (BERT) by modifying some key hyperparameters, training with much larger learning rates and batches, and removing BERT's next-sentence pretraining objective (Liu et al., 2019). Critically, this allows RoBERTa to improve on the masked language modeling objective and allows for better task performance down the road. The RoBERTa base is composed of 12-layers, 768-hidden, 12 self-attention heads, and 125M parameters.

Processing each input tweet first began with changing sarcasm labels from 1 (sarcastic) to 0.8 and 0 (non-sarcastic) to 0.2, although the 0.2 was eventually changed back to 0. This was to account for random noise in the dataset and for training examples that were of lesser quality than others.

The next step was tweet normalization: among other things, this included replacing hyperlinks, user tags, and emojis with a standardized token (`"@URL"` for hyperlinks and `"@USER"` for user tags, for example). Furthermore, contracted words were separated out to extract the key token. This was to ensure the model did not learn from random, irrelevant noise found in hyperlinks or user tags.

Before passing each normalized tweet into the tokenizer, however, we first extracted all verbs and adjectives from the original tweet (Sequence A) and strung them together with whitespace to create a new string (Sequence B). In turn, those verbs and adjectives were replaced with the `<mask>` token in the original tweet. This was executed in an attempt to help the model better learn the relationship between a tweet's sarcastic presence and any available verbs or adjectives in it. These two sequences were then joined together with separator tag `</s>` and fed into the tokenizer as a single sequence.

Padding tokens were added to make each input the same length for the RoBERTa model. The maximum length used for this system was 256. Although the longest string of tokens from the available training dataset was 111, we set it to 256 for the sake of safety. This ensured that all tensor inputs were set to equal the maximum sequence length used for batched parallelized training. This meant the ultimate input passed into the tokenizer looked like Table 1, where `<s>` is the classifier token, `</s>` is the separator token, `<pad>` is the padding token, `Seq-A` contains `<mask>` tokens where its adjectives and verbs originally were, and `Seq-B` is a string of all verbs and adjectives from the original tweet.

Input	`<s>Seq-A</s>Seq-B<pad><pad>`

Table 1: A sample input for encoding.

A sequence classification head containing a linear layer was applied on top of the final hidden-states output, with a label prediction of 1 denoting a sarcastic tweet and 0 denoting a non-sarcastic tweet. For standalone tweets (Task A), the threshold to pass a tweet as sarcastic was set to 0.40, where tweets with a score higher than that were marked as sarcastic while those underneath this threshold were not. For determining which of two tweets is sarcastic (Task C), the tweet with the highest absolute score, regardless of its relation to a threshold, was marked as sarcastic.

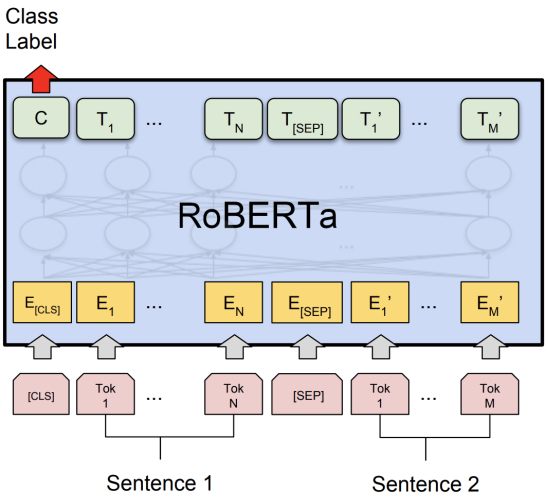

Figure 1: The RoBERTa model architecture.

2.2 XLM-RoBERTa for Binary Classification

The Arabic subtask for Tasks A and C require us to distinguish whether a tweet is sarcastic, either as a standalone text (Task A) or in comparison to another tweet (Task C). We framed these as binary classification problems and leveraged the XLM-RoBERTa$_{BASE}$ model, which is a multi-lingual version of the RoBERTa model and is pre-trained on 2.5TB of filtered CommonCrawl data containing 100 languages (Conneau et al., 2019). It is composed of 12-layers, 768 hidden, 8 self-attention heads, and 125M parameters.

In preprocessing each input Arabic tweet, we began by changing the sarcasm confidence labels from 1 to 0.8. This was again executed to account for random noise as well as subpar training data examples that did not encapsulate sarcastic qualities nearly as well as others.

Regular tweet normalization does not apply to Arabic. Certain qualities in the written form of Arabic, such as diacritization, further complicate this matter. The same word in two different diacritizations can have meanings that are seemingly completely unrelated (Alkhatib, 2017). This introduces difficulty in extracting the true semantic meaning of a text in Arabic.

We therefore relied on CAMeL Tools, a Python library designed for the Arabic language to execute dediacritization and remove any non-essential components from the input texts (Obeid et al., 2020). Further normalization was also conducted with functions from this specialized library, such as removing orthographic ambiguity.

After processing input tweets, the technique of extracting verbs and adjectives to form a secondary sequence was utilized. As in the first approach, the extracted verbs and adjectives were replaced with `<mask>` tokens in the original input tweet, and this was prepended to the secondary sequence of verbs and adjectives using a `</s>` separator token. This combined sequence was then fed into the XLM-RoBERTa's tokenizer to be encoded. In this particular case, a specialized part-of-speech tagger was used from CAMeL Tools to identify and extract each input tweet's set of verbs and adjectives.

During tokenization, padding tokens were added to the right to make all the tensor inputs of uniform length. Although the longest examined length of any tweet was 143, we utilized 256 to account for any unexpected inputs.

The final element of this system was a sequence classification head containing a linear layer that was applied on top of the final hidden-states output with a label of 1 predicting sarcasm and a label of 0 predicting otherwise. For detection of sarcasm in standalone Arabic tweets (Task A), those with a score that exceeded our threshold of 0.40 were marked as sarcastic while others were not. For determining the sarcastic tweet in a pair of tweets (Task C), the tweet with the highest absolute score, regardless of whether it exceeded Task A's threshold of 0.40, was marked as sarcastic, while the other tweet was not.

2.3 BERT Base for Multilabel Classification

Task B required us to distinguish which category of sarcasm an English tweet belonged to, of which there are six: general sarcasm, irony, satire, understatement, overstatement, and rhetorical question. A tweet can belong to none, one, or multiple categories. We framed this task as a multilabel sequence classification problem and we leveraged the BERT$_{BASE}$ model (see Figure 2 for a high-level architecture visual). It is composed of 12-layers, 768-hidden, 12 self-attention heads, and 110M parameters (Devlin et al., 2018).

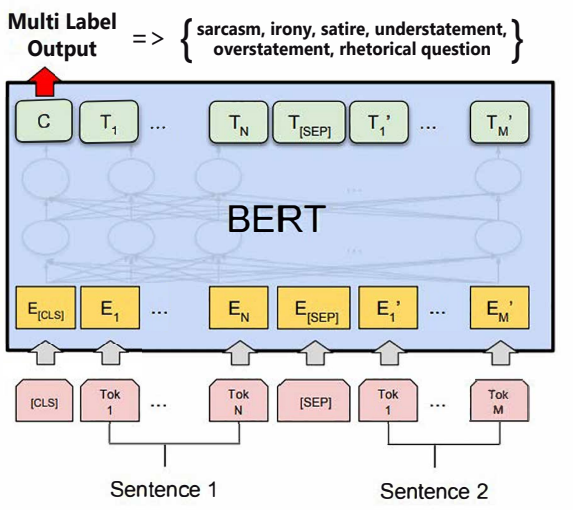

Figure 2: The BERT model architecture modified to reflect multilabel output.

Each tweet was first normalized following the steps outlined in the first approach. This included "demojifying" any present emojis and standardizing any present hyperlinks and user tags with `"@URL"` and `"@USER"`. Before feeding the tweet into the tokenizer, however, we utilized an approach to improve the model's understanding of the keywords that might point towards a particular

Data Augmented Results	Scores		
	Weighted Recall	Weighted Precision	Macro-F1 Score
Baseline (No Augmentation)	0.549	0.091	0.156
Category Duplication	**0.642**	0.089	0.156
Manual Sentence Generation	0.561	**0.093**	**0.159**
GPT-3 Sentence Generation	0.552	0.082	0.151

Table 2: A summary of the performance of data-augmented systems for multilabel classification. The highest performing technique is bolded for weighted recall, weighted precision, and macro F1 score.

category of sarcasm. A TF-IDF vectorizer was used to extract the 15 most significant and representative keywords across sentences of each category. Then, for each input training tweet, all the keywords associated with the categories of sarcasm the tweet belonged to were strung together to create a secondary sequence (Sequence B) that was appended to the original input tweet (Sequence A) and separated with a separator token (</s>). This combined sequence was then fed into the $\text{BERT}_{\text{BASE}}$ tokenizer as a whole. This was done in an attempt to help the model be able to better seek out key phrases and words that might indicate a tweet's categorization as sarcastic, ironic, satirical, etc.

After training the multilabel classifier and generating six predictions for a tweet's likelihood of categorization in each of the six sarcastic categories, the tweet would be marked as valid for a category if its score was greater than 0.30. This was a relatively low threshold that we felt was necessary to account for the similarities across tweets belonging to different categories of satire. Furthermore, we needed to compensate for the lack of training data in categories like satire, understatement, and overstatement, which had 25, 10, and 40 training examples, respectively. By setting a lower threshold, we are able to ensure that we are not prohibitively preventing classifying any tweets as satire, understatement, overstatement, or any other category.

2.3.1 Data Augmentation

To support our efforts for multilabel classification, we explored three data augmentation techniques.

Our first technique was simply duplicating the satire, understatement, and overstatement categories to double the quantity of sentences in each of those categories. This involved copying and pasting each sentence back into the category to hopefully strengthen the model's understanding of sarcastic keywords, phrases, and qualities. We observed no meaningful improvement in results.

The second technique involved the manual generation of sentences to expand the dataset. As mentioned before, a TF-IDF vectorizer was used to extract the 15 most relevant and representative keywords for sentences across each category. See Figure 2 for some example keywords extracted through this technique. Using basic sentence templates, new training data examples were created with keywords for each satirical category being substituted into various parts of each new training datapoint. 30-40 new training examples were created for each of the satire, understatement, and overstatement categories. However, this effort did not yield meaningful or significant improvement in results.

Sarcasm	"just", "like", "really"
Understatement	"good", "like", "sorta"
Overstatement	"hate", "love", "worst"

Table 3: A subset of keywords observed as the most representative for sarcasm and under/over-statements through a TF-IDF vectorizer.

A final technique involved utilizing GPT-3 for generating new sentences. A prompt like "Generate 10 sarcastic sentences" or "Create 15 rhetorical questions" was provided to the model, however it was observed that the produced sentences were particularly repetitive with little variance in structure or style. The difference between most sentences produced by the model was a simple substitution in topic, object, or subject, especially as we asked the model to produce an increasing number of new sentences. A lack of training data to properly fine-tune the GPT-3 model was likely an issue here, and this technique was eventually dropped.

Results for each of these techniques are provided in Table 2, with the highest performing technique bolded for each metric. As observed, while most techniques seemed to perform better than the base-

line, the improvements are marginal and unreliable.

2.3.2 Additional Techniques

It is worth noting that another system was explored to conduct multilabel classification. Specifically, we attempted to create 6 binary classifiers (one for each category of sarcasm) with the intent of aggregating results across all binary classifiers to mimic a multilabel classifier's output. This, however, was complicated by the severe lack of training examples for some categories as well as issues with computational capacity and consumption on Google Colab Pro. This system was eventually dropped in favor of a single multilabel classifier built with $\text{BERT}_{\text{BASE}}$, as described earlier.

3 Experimental Setup

3.1 Dataset

The task provided two training data files, one for English and another for Arabic. In each case, the task organizers provided the sarcasm labels for each tweet themselves. This avoided the need to rely on existing proxies like predefined tags or third-party labelers (Abu Farha et al., 2022).

Within the English training file, there are nine pieces of information: the tweet, a `0/1` value for the presence of sarcasm, a non-sarcastic rephrase of that tweet, and a `0/1` value for each of the various sarcasm subtypes (general sarcasm, irony, satire, understatement, overstatement, and rhetorical question). This dataset contained 3466 training examples, of which 866 were sarcastic and the remaining 2600 were not. These 866 examples are further split into multiple labels as follows: 713 for sarcasm, 155 for irony, 25 for satire, 10 for understatement, 40 for overstatement, and 101 for rhetorical question. The underresourced nature of categories like satire, understatement, and overstatement introduced challenges for our multilabel classifier system in extracting and understanding the key characteristics belonging to those categories. It is worth noting that data quality is, at times, questionable, with training examples such as *"whoop diddy scoop poop"* adding random noise into an already scarce dataset.

Within the Arabic training file, there are four pieces of information: the tweet, a `0/1` value for the presence of sarcasm, a non-sarcastic rephrase of the tweet, and a dialect label for that particular tweet (e.g. Nile, Maghreb). This training file contained 3102 training examples, of which 745 were sarcastic and the remaining 2357 were not.

3.2 Evaluation Metric

For both the English and Arabic binary classification approaches, a confusion matrix was produced to determine accuracy, precision, recall, and F-1 scores. For the multilabel classification task, weighted precision/recall for each category as well as macro F-1 scores were utilized.

3.3 Implementation Details

All three systems were developed with the PyTorch framework, HuggingFace's transformers library for the BERT, RoBERTa, and XLM-RoBERTa models, and Google Colab Pro using a single Tesla P100-PCIE-16GB GPU. For Tasks A and C, the English and Arabic binary classifiers trained for those problems shared the same hyperparameters: training and validation batch sizes of 16, a maximum sequence length of 256, 6 training epochs, and an AdamW optimizer with a learning rate of 3e-5 and epsilon value of 1e-8. For Task B, the English multilabel classifier's training policy utilized the following hyperparameter values: training and validation batch sizes of 16, a maximum sequence length of 256, 4 training epochs, and an AdamW optimizer with a learning rate of 1e-05 and epsilon value of 1e-12. All other hyperparameter values were set to their defaults according to the HuggingFace implementation. It should also be noted that for all systems, a random seed was set for the sake of reproducibility.

4 Experimental Results

Team Dartmouth received the following ranks:

- Task A (English): 13th place
- Task A (Arabic): 9th place
- Task B: 14th place
- Task C (English): 12th place
- Task C (Arabic): 10th place

Table 4 on the following page displays a tabular summary of all official scores received on each of the five subtasks, which vary from accuracy and precision to recall and macro F1 scores. As observed, our best performing system for binary classification was the English classifier developed for Task A, whereas our worst performing for binary

Binary Classification Tasks	Scores				
	F-1 Sarcastic	F1-Score	Precision	Recall	Accuracy
Task A (English)	0.386	0.635	0.625	0.648	**0.804**
Task A (Arabic)	0.350	0.529	0.581	0.665	0.597
Task C (English)	-	0.659	-	-	0.660
Task C (Arabic)	-	0.679	-	-	0.680

Multilabel	Scores						
	Macro F1	F1-Sarcasm	F1-Irony	F1-Satire	F1-Under	F1-Over	F1-Rhet-Q
Task B	0.0590	0.2293	0.0202	0.0824	0.0000	0.0077	0.0143

Table 4: A tabular summarization of the performance of all three systems across all five subtasks, reporting various metrics including accuracy, precision, recall, and regular/macro F-1 scores. Experiments revealed that further data augmentation did not improve the scores of any system.

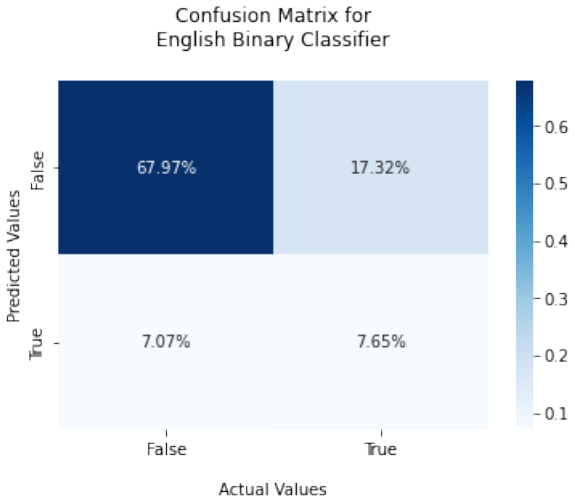

Figure 3: A confusion matrix of the English binary classifier developed for Tasks A and C.

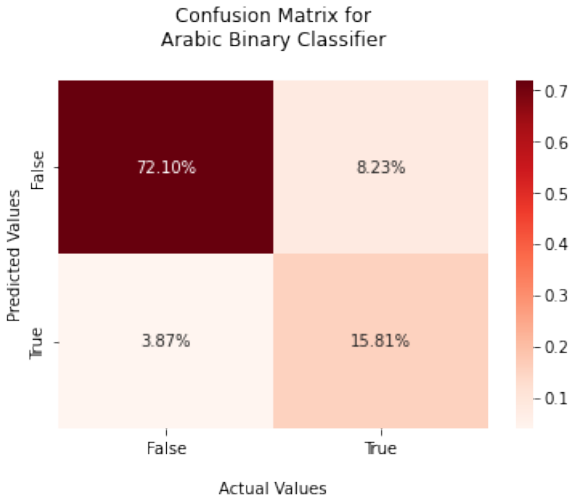

Figure 4: A confusion matrix of the Arabic binary classifier developed for Tasks A and C.

classification was the Arabic classifier also developed for Task A. For the multilabel classification task, macro F-1 as well as weighted categorical scores are provided.

Confusion matrices displaying our best development metrics are provided in Figures 3 and 4. This reveals insights into the relatively imbalanced split of the training dataset, creating issues which were further compounded by the overall small number of training examples.

4.1 Case Study: Task A (English)

It's worth exploring Task A (English) in greater detail to understand the elements that factored into our model's scores. To begin, it should be noted that certain inputs in the test dataset for this subtask were sometimes single tweets like "Followed" or "Pinball!", while other tweets were random noise, such as the following:

> "20:00 GMT:
> Temp: 13.7°C,
> Wind: SSW, 3 mph (ave), 8 mph (gust),
> Humidity: 92
> Rain (hourly) 0.0 mm,
> Pressure: 1017 hPa, rising slowly."

The prevalence of random noise such as the above in the test set can make it somewhat challenging for the model at hand to be able to relate the test input to what it has learned. There's very little context to learn from in one-word tweets like the ones mentioned, and this may bias the model towards marking such tweets as non-sarcastic, when in reality they may be sarcastic (e.g. perhaps the one-word tweet was a sarcastic remark towards another tweet). Granted, this is the nature of text in

short-form online social forums like Twitter, but it does contribute to a concrete decrease in model performance.

Furthermore, our technique of masking verbs and adjectives in the original tweet while simultaneously stringing those verbs and adjectives together into a second sequence to be fed into the tokenizer alongside the tweet input may have overfit the model towards certain words and phrases from the training dataset. While this may have been helpful in identifying sarcastic tweets which did include those words, it may also have caused the model to overlook other sarcastic sentences that did not include them in the test dataset.

As such, random noise, poor quality, and certain learning techniques may have been factors in contributing to the scores received by our binary and multilabel classification systems. The same observations apply to Tasks A and C in Arabic as well as Task B. In particular, our technique of extracting and appending the top 15 words for each category a tweet belongs to may have inadvertently overfit the model to overlook other textual signals that indicate a tweet's sarcastic categorization in preference for certain words and phrases.

5 Conclusion and Future Work

In this paper, we have described three binary and multilabel sequence classification systems using the BERT, RoBERTa, and XLM-RoBERTa architectures from HuggingFace for the detection of sarcasm in English and Arabic tweets. We found that additional work to augment the training data with duplication of sentences and manually/automatically synthesizing new sarcastic sentences did not improve the results of the model. Furthermore, challenges were observed with the multilabel classifier in learning to extract the key characteristics that categorize a tweet as a distinct example of satire, understatement, or overstatement – categories which were generally underresourced in the training dataset.

Further investigation could include implementing six binary classifiers instead of a single multilabel sequence classifier for Task B. Given enough training time, data, and resources, it could certainly be the case that aggregating results across specialized binary classifiers provide more concrete results than what has been produced. In particular, this may allow each of the binary classifiers to more deeply learn the unique characteristics, keywords, and structure of the sarcastic sentences it ingests.

It would also be interesting to see how the results across all three systems change with sufficient training data, with perhaps tens of thousands of more valid examples that can allow the models to truly capture the essence of sarcasm across a wide and varied set of training examples.

6 Acknowledgements

Rishik Lad is grateful to be supported by the James O. Freedman Presidential Scholars program at Dartmouth College.

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Manar Alkhatib. 2017. *Challenges in Arabic Natural Language Processing*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/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. *CoRR*, abs/1907.11692.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL tools: An open source python toolkit for Arabic natural language processing. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 7022–7032, Marseille, France. European Language Resources Association.

(Liu et al., 2019) (Conneau et al., 2019) (Abu Farha et al., 2022) (Obeid et al., 2020) (Alkhatib, 2017) (Devlin et al., 2018)