Parameter and Data Efficient Continual Pre-training for Robustness to Dialectal Variance in Arabic

Soumajyoti Sarkar **Kaixiang Lin** **Sailik Sengupta**

Leonard Lausen **Sheng Zha** **Saab Mansour**

aws ai Labs
{soumajs,kaixianl,sailiks,lausen,zhasheng,saabm}@amazon.com

Abstract

The use of multilingual language models for tasks in low and high-resource languages has been a success story in deep learning. In recent times, Arabic has been receiving widespread attention on account of its dialectal variance. While prior research studies have tried to adapt these multilingual models for dialectal variants of Arabic, it still remains a challenging problem owing to the lack of sufficient monolingual dialectal data and parallel translation data of such dialectal variants. It remains an open problem on whether the limited dialectical data can be used to improve the models trained in Arabic on its dialectal variants. First, we show that multilingual-BERT (mBERT) incrementally pretrained on Arabic monolingual data takes less training time and yields comparable accuracy when compared to our custom monolingual Arabic model and beat existing models (by an avg metric of +6.41). We then explore two continual pre-training methods– (1) using small amounts of dialectical data for continual finetuning and (2) parallel Arabic to English data and a Translation Language Modeling loss function. We show that both approaches help improve performance on dialectal classification tasks ($+4.64$ avg. gain) when used on monolingual models.

1 Introduction

Pre-trained language models such as BERT [1] have been the backbone of many classification systems processing textual inputs. The two-step procedure for training these models is to first pre-train a language model $\mathcal{M}$ on some data followed by addition of a classification layer on top and finetuning $\mathcal{M}$ on a smaller target classification task data. In multilingual scenarios, transformer-based architectures are trained on a large multilingual corpora at the pretraining stage [1–3]. Similar to mono-lingual models, they are then finetuned on a task specific to languages present in the pretraining data. The assumption that the finetuning task data has language overlap with the pretraining data is important; in zero-shot settings, continued pretraining on the new language is often necessary to improve model performance [4].

In this paper, we seek to understand the distinctions between developing monolingual models from scratch and continual pretraining of multilingual models for Arabic, a language known to have rich dialectical variance. While work exists on developing multilingual models robust to realistic multilingual noise [5], they remain ignorant to dialectical diversity. Modern Standard Arabic (MSA) has grained popularity at an international stage, dialectical variants like Egyptian Arabic, Levantine Arabic, and Moroccan Arabic are also widely used [6]. Adapting to these dialectal variants is often non-trivial owing to differences between the dialects that may range from a syntactic (vocabulary,

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

morphology) to a semantic one (same phrase could have different formality, context, or meaning) [7]. This makes the task of developing Natural Language Systems for Arabic a challenging one. In light of these challenges, this paper discusses applying language model pretraining and incremental learning to tackle this resource deficiency of dialectal Arabic data. We expect our work to help motivate future work on developing models for dialectical Arabic and languages with dialectical diversity.

To complicate matters further, there is a distinct scarcity of large corpora for the dialectical variants in Arabic. It is not immediately evident if development of monolingual models vs. adaptation of multilingual models are the right path forward for scenarios where a language (in our case, Arabic) demonstrates dialectical diversity. For this purpose, we leverage and compare our models to publicly available monolingual araBERT [8] and multilingual BERT (mBERT) [1] models. To ensure our improvements don't come at the cost of impacting existing model performance, our benchmarks encompass both non-dialectical and dialectical tasks.

Our paper is structured as follows: we start off with describing the benchmarks, i.e. the downstream tasks that evaluate the different baseline and developed models on Arabic language and its dialectal variants (§2). We then focus on discussing methods to build our model, elucidating the model choices, the pretraining corpora and the objectives for adapting to dialectal data (§3). Finally, we demonstrate the performance of the various pretrained models on the downstream tasks.

2 Downstream Task Benchmarks

In order to evaluate the pretrained Arabic and multilingual models on dialectal variants, we use the eight tasks proposed in the ALUE benchmark [9]. This recent benchmark curates a diversified collection of Arabic NLU tasks that could be categorized as follows:

1. *Single Sentence Classification*: It consists of 4 tasks on dialectal Arabic data - MADAR Shared Task Subtask 1 (MDD) which classifies sentences into their dialects based on cities, OSACT4 Shared Task on Offensive Language Detection (OOLD), where a tweet is labeled offensive if it consists of inappropriate language & OHSD where offensive tweets from the same task are classified as hate speech based on a different criteria) and IDAT@FIRE2019 Irony Detection Task (FID).
2. *Sentence Pair Classification*: It consists of 2 tasks on NSURL-2019 Shared Task 8 (MQ2Q) in which a pair of questions is assumed to be classified to be similar if they have the same exact answer and meaning and Cross-lingual Sentence Representations (XNLI) in which sentence pairs are labeled with either one of the following logical relationship labels: entailment, contradictory, or neutral.
3. *Multi-label classification*: SemEval-2018 Task 1 - Affect in Tweets, Emotion Classification task (SEC) on dialectal data.
4. *Regression*: Sentiment Intensity Regression task (SVREG) on dialectal data.

Six out of the eight ALUE tasks contains dialectical data, of which five are constructed with twitter data, which by its very nature, has dialectical variance. This makes ALUE a suitable choice of evaluating our models since we experiment with adaptation to dialectal data. More information on the ALUE data can be found in Appendix Section A.3.1. We use the code script provided by the researchers [1]. Across the various tasks, we present Accuracy for XNLI, Pearson for SVREG and Jaccard for SEC, and F1-score for the other tasks. We use the same hyper-parameters used in the setup in [9], and we finetune for 10 epochs with a maximum sequence length of 256 for all tasks.

3 Language Model Pretraining

As mentioned earlier, we set out on training our own Arabic language model that exhibits robustness to dialectal variants. Owing to strict memory and inference-time constraints, we could not consider a BERT-Base (12-layer) model. Further, owing to limited task-time training data, we found a 4-layer transformer architecture to be the best-suited for our needs. For our comparison with baseline models, we truncate the publicly available BERT-Base monolingual AraBERT[2] and mBERT models by

[1] https://github.com/Alue-Benchmark/alue_baselines

[2] https://github.com/aub-mind/arabert

Code	Corpora	Size
C1	Oscar Arabic	67M
	Arabic Wiki	49M
	Arabic CC100	111M
	Arabic Newswire Part-1	2.3M
	Arabic Gigaword Fifth Edition	96M
	Gulf Arabic Conv.	4K
	GALE (only Arabic data)	25K
	BOLT SMS/Chat (only Arabic data)	44K
C2	OSCAR Egyptian Arabic Corpus	102K
C3	GALE Parallel Corpus BOLT Egyptian Arabic SMS/Chat	255K

Table 1: The number of sentences present in each corpora after a token-hashing heuristic was used to remove sentences with high duplicate tokens.

keeping only the first 4 layers of the 12 layer pretrained model. Although it has been known that different layers in BERT models capture different kinds of information [10], we also verified that truncating these publicly pretrained baseline models to 4-layers followed by task level fine-tuning did not degrade their performance significantly (as captured by the metrics in our benchmark compared to what is reported in the papers) while improving the training efficiency, inference latency, and memory requirements (as parameters reduce by one-third).

Exploring the setup of incremental learning with a 4 layer model instead of a 12 layer model allows us to measure the aspect of model adaptation to low resource dialectal data. We believe that any affirmation that smaller models can adapt to small amounts of dialectal Arabic even when pretrained on the dominant MSA data, could work as an indicator for a similar pattern for larger models on more data which we leave as future work. In addition to the publicly available AraBERT model, we explored training our own model from scratch due to (1) more independence in using pretrained dialectical data (eg. importance sampling, data-duplication strategies etc.) and (2) commercially restrictive licensing of AraBERT models. The main questions we set out to answer in the course of training these models are:

- RQ1 Will continually pretraining the multilingual models for fewer epochs bring comparable performance to a monolingual model trained from scratch when both are trained on the same corpora and the tokenizer vocab remains unchanged?
- RQ2 Given more dialectical data, does cost-effective continual finetuning of multilingual model perform at par with continual finetuning of monolingual models?
- RQ3 Can we leverage a parallel-data corpora to further boost model performance on?

3.1 Pretraining corpora

We use three different sets of pretraining corpora for the three stages of model training (in §3.2):

[C1] Mixed Arabic and its dialectal variants The set composed of various Arabic corpora of different sizes as shown in Table 1. Of these, the Arabic Wiki 2020 snapshot contains mixed English words as well as some transliterated sentences apart from the Arabic sentences. We use the Arabic source data of the GALE Arabic parallel data and the BOLT Egyptian Arabic SMS/Chat data, which is significantly smaller in size compared to the other corpora. We have ≈326 million sentences from these sources and use it to train models from scratch and continually pretrain m-BERT.

[C2] Egyptian Arabic data We use the 2019 Egyptian Arabic dialectal corpora from Oscar; it contains 102K sentences. This data is used to continually pretrain a model already trained on *C1* and gauge whether its inclusion helps in improving performance on ALUE.

[C3] Parallel Data We use parallel data ($\approx 255K$ sentences) on dialectical variants to English from the two GALE Arabic Parallel datasets, which contains transcripts for various news and

Model	FID	MDD	MQ2Q	SVREG	SEC	OOLD	OHSD	XNLI
AraBERT	78.31	51.15	77.41	42.41	32.21	94.92	96.57	51.02
AraBERT-Twitter	79.73	52.26	77.07	39.25	31.34	94.21	97.76	39.71
mBERT	77.14	49.31	77.11	34.70	35.49	94.13	96.49	**51.08**
m-B-Ar	79.61	**56.04**	80.26	50.82	41.05	94.62	97.13	50.57
B-Ar	79.32	55.84	**80.35**	51.65	41.88	94.58	97.27	51.04
t-B-Ar	**81.04**	53.49	72.63	**74.37**	**49.26**	**95.12**	**98.36**	51.03

Table 2: Our models (last 3 rows) outperform existing models on all ALUE task metrics (Pearson for SVREG, Jaccard for SEC, accuracy for XNLI, and F1-score for the rest) except one. The columns in green denote benchmarks with dialectical data. All these models have 4 layers in the BERT style architecture.

conversations television programs broadcast in different dialects [11, 12], and the BOLT Egyptian Arabic SMS/Chat dataset, which contains manual translations of messages in Egyptian Arabic [13].

For deduplicating the sentences, we use a simple approximation method on each corpus separately in Table 1. We space tokenize each sentence and for rolling windows of size 10 (in units of tokens) in a sentence, we compute the MD5 hash of each of these windows. Using these hash values, we compute the frequency of all such windows in all the sentences in a corpus. Following this, for each sentence, we compute the ratio of windows in a sentence which have been repeated more than one in the entire corpus. If that ratio is sufficiently high (we set that to 0.7), we remove that sentence from the corpus. To make sure that we do not remove all candidates among a pool of exactly duplicate sentences (the ratio would be 1 in that case for the exactly duplicate sentences), we keep one among the pool and remove the rest.

3.2 Model evolution

All models we trained are encoder models, specifically the BERT base architecture. As stated above, owing to resource and efficiency constraints, we use 4 layers for purposes of our experiments which amount to a total of 11M parameters. We use a global batch size of 1024 and a learning rate of 1e-5 in a multi-node cluster setup. Details on the hyperparamters for the pretraining is discussed in Appendix Section A.2. The first three models we trained were the following:

m-B-Ar Multilingual BERT (mBERT) continually pretrained for 800K steps starting from the pretrained public checkpoint

B-Ar 4-layer BERT model trained from scratch (with randomly initialized weights) with C1 for 1.3M steps

t-B-Ar B-Ar model trained with a custom tokenizer trained on C1

These three models were reasonably straightforward choices to compare with the publicly available AraBERT model[3]. We trained m-B-Arfor 800K steps and B-Arfor 1.3M steps as we found the models started to overfit on the data as evidenced by the increase in validation losses after these many steps on the respective models. We further consider a version of AraBERT trained on Twitter Data (AraBERT-Twitter)[4] that we believe will exhibit dialectical robustness consider the dialectically diverse twitter corpora. For m-B-Ar and B-Ar , we use off-the-shelf multilingual bert tokenizer with the cased version of the vocab. For all the three models, we use the Masked Language Modeling (MLM) objective at the pretraining stage and discard the Next Sentence Prediction (NSP) unsupervised task. For t-B-Ar, we trained a word-piece tokenizer with vocabulary size of 64K. We present the results in Table 2. We clearly see that in this resource and latency constrained setting, our models outperform all of the existing models– t-B-Ar's avg. metric of 71.91 is significantly higher compared to the best-performing model AraBERT with an avg. metric of 65.50. We find that m-B-Ar and B-Ar perform similar to each other; 68.76 $vs.$ 68.99 on avg. across all tasks. With 128 NVIDIA

[3] AraBERTv0.2-base: https://huggingface.co/aubmindlab/bert-base-arabert

[4] AraBERTv0.2-twitter: https://huggingface.co/aubmindlab/bert-base-arabertv02-twitter

Model	FID	MDD	MQ2Q	SVREG	SEC	OOLD	OHSD	XNLI
m-B-Ar	79.61	56.04	80.26	50.82	41.05	94.62	97.13	50.57
+C2	79.35	56.60	80.46	53.72	40.13	94.56	97.16	51.33
+C2+C3	78.29	56.82	80.65	51.42	40.75	**95.18**	97.51	**52.69**
B-Ar	79.32	55.84	80.35	51.65	41.88	94.58	97.27	51.04
+C2	**81.20**	55.84	84.73	69.72	47.66	94.53	97.75	52.13
+C2+C3	79.9	**57.61**	**85.31**	**70.31**	**48.03**	94.67	**97.91**	51.38

Table 3: Effectiveness of dialectical and parallel data for continual pretraining.

A100 GPUs, m-B-Ar takes approximately 60% of the time it takes to train B-Ar to convergence keeping all hyper-parameters the same. Hence, RQ1 holds.

3.3 Training with Dialectal and Parallel Data

While corpora C2 and C3 can be used along with C1, we observed higher gains in metrics when the dialectical data was used in the following ways:

+C2 Continual pre-training with the dialectical corpora C2 and MLM loss.

+C2+C3 Continual pre-training of the above model using parallel data in C3 and a Translation Language Modeling (TLM) loss [2] where the two parallel sentences are concatenated and the position embedding is reset for the target sentence.

Owing to Out-Of-Vocabulary issues on a smaller fixed vocabulary, we ignore t-B-Ar for experiments in Table 3. To answer RQ2, we show results of incrementally fine-tuning m-B-Ar and B-Aron dialectical data in Table 3. First, we note that the performance of the m-B-Ar model hardly shows any improvement when dialectical data is used ($68.76 \rightarrow 69.16$ avg) compared to a noteworthy improvement seen on B-Ar ($68.99 \rightarrow 73.14$ avg). B-Ar outperforms m-B-Ar models on six tasks and is within one F1-score/accuracy on the other two. Hence, RQ2 does not hold; continual finetuning of monolingual models trained on more dialectical data outperforms finetuning of multilingual models.

To answer RQ3, we noticed that the improvements are negligible when C3 and TLM loss is used for the multilingual m-B-Ar ($69.16 \rightarrow 69.17$) and small when used on top of the monolingual B-Ar ($72.94 \rightarrow 73.14$). On the six dialectical tasks, the improvement disappears for m-B-Ar ($70.25 \rightarrow 69.96$) and slightly improves for B-Ar ($74.45 \rightarrow 74.73$). On the dialectical benchmarks, we see gains when using B-Ar+C2+C3 on almost all tasks (esp. high gains seen on MDD) but FID where a steep degradation averages out most of the gains seen. While further investigation and approaches to use the parallel corpora C3 is needed, we could not strongly conclude that use of a parallel data corpora was effective in boosting model performance.

4 Conclusion

In this paper, we are able to show that existing multilingual models can be drastically improved with continual finetuning outperforming publicly available monolingual models on the ALUE benchmark. We observe that low-resource dialectical data can further improve performance of monolingual models, but don't have a pronounced effect on multilingual models. Unfortunately, parallel corpora data and Translation Language Modeling loss that can leverage this data does not yield further improvements. For future work, we also plan to investigate advanced alignment techniques and better leverage the parallel corpora data in the future. Additionally, while there is a broad range of studies in the area of adversarial training that could be applied to dialectal data, another area is to craft adversarial training examples using augmentations prior to training that can help adapt easily to dialectal data. In this paper, we considered Arabert as the baseline model for our comparison as it has variants trained on MSA as well as dialectal data and has been considered a robust baseline model until recently and in future, we plan to continue this work with comparisons on more recent SOA models on Arabic data which have been publicly released.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

[2] Alexis CONNEAU and Guillaume Lample. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf.

[3] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

[4] Yoshinari Fujinuma, Jordan Boyd-Graber, and Katharina Kann. Match the script, adapt if multilingual: Analyzing the effect of multilingual pretraining on cross-lingual transferability. *arXiv preprint arXiv:2203.10753*, 2022.

[5] Asa Cooper Stickland, Sailik Sengupta, Jason Krone, Saab Mansour, and He He. Robustification of multilingual language models to real-world noise with robust contrastive pretraining. *arXiv e-prints*, pages arXiv–2210, 2022.

[6] Omar F Zaidan and Chris Callison-Burch. Arabic dialect identification. *Computational Linguistics*, 40(1):171–202, 2014.

[7] Nizar Habash, Ryan Roth, Owen Rambow, Ramy Eskander, and Nadi Tomeh. Morphological analysis and disambiguation for dialectal arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 426–432, 2013.

[8] Wissam Antoun, Fady Baly, and Hazem Hajj. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*, 2020.

[9] Haitham Seelawi, Ibraheem Tuffaha, Mahmoud Gzawi, Wael Farhan, Bashar Talafha, Riham Badawi, Zyad Sober, Oday Al-Dweik, Abed Alhakim Freihat, and Hussein Al-Natsheh. Alue: Arabic language understanding evaluation. In *Proceedings of the Sixth Arabic Natural Language Processing Workshop*, pages 173–184, 2021.

[10] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[11] Xiaoyi Ma, Dalal Zakhary, and Stephanie Strassel. *GALE Phase 1 Arabic Broadcast News Parallel Text - Part 1 LDC2007T24. Web Download*. Linguistic Data Consortium, Philadelphia, 2007.

[12] Xiaoyi Ma, Dalal Zakhary, and Stephanie Strassel. *GALE Phase 1 Arabic Broadcast News Parallel Text - Part 2 LDC2008T09. Web Download*. Linguistic Data Consortium, Philadelphia, 2008.

[13] Jennifer Tracey et al. *BOLT Egyptian Arabic SMS/Chat Parallel Training Data LDC2021T15. Web Download*. Linguistic Data Consortium, Philadelphia, 2021.

[14] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

A Appendix

A.1 Pretraining details

A.1.1 Corpus

The public links to the corpus used is below:

1. OSCAR - https://oscar-corpus.com/post/oscar-2019/
2. Arabic Wiki - https://archive.org/details/arwiki-20190201
3. Arabic CC100 - https://data.statmt.org/cc-100/
4. Arabic Gigaword Fifth Edition - https://catalog.ldc.upenn.edu/LDC2011T11
5. Gulf Arabic Conversational Telephone Speech, Transcripts - https://catalog.ldc.upenn.edu/LDC2006T15
6. GALE Phase 1 Arabic Broadcast News Parallel Text - Part 1 - https://catalog.ldc.upenn.edu/LDC2012T18
7. GALE Phase 1 Arabic Blog Parallel Text - https://catalog.ldc.upenn.edu/LDC2008T02
8. GALE Phase 1 Arabic Broadcast News Parallel Text - Part 2 - https://catalog.ldc.upenn.edu/LDC2008T09
9. GALE Phase 1 Arabic Newsgroup Parallel Text - Part 1 - https://catalog.ldc.upenn.edu/LDC2009T03
10. GALE Phase 1 Arabic Newsgroup Parallel Text - Part 2 - https://catalog.ldc.upenn.edu/LDC2009T09
11. GALE Phase 2 Arabic Broadcast Conversation Parallel Text Part 1 - https://catalog.ldc.upenn.edu/LDC2012T06
12. GALE Phase 2 Arabic Broadcast Conversation Parallel Text Part 2 - https://catalog.ldc.upenn.edu/LDC2012T14
13. GALE Phase 2 Arabic Broadcast News Parallel Text - https://catalog.ldc.upenn.edu/LDC2012T18
14. GALE Phase 2 Arabic Web Parallel Text - https://catalog.ldc.upenn.edu/LDC2013T01
15. BOLT Egyptian Arabic SMS/Chat and Transliteration - https://catalog.ldc.upenn.edu/LDC2017T07
16. BOLT Egyptian Arabic SMS/Chat Parallel Training Data - https://catalog.ldc.upenn.edu/LDC2021T15 Arabic Online Commentary Dataset - https://aclanthology.org/P11-2007/

For processing the text, we take inspiration from some of the procedures adapted in AraBERT [5] which includes removing emoji, tashkeel, tatweel,and html markup. Additionally, we also used some more steps to clean the Arabic wiki corpus. A lot of the sentences in the Arabic Wiki corpus contained templates from the page that specified the author name, the topic which did not contribute much to meaningful sentences. We removed such sentences using a combination of keyword search and regex expressions. We did not apply the Farasa pre-segmentation as done in [8] in any of corpus used to train our models.

A.2 Model Training

We perform pre-training on 16 nodes, each with 8 NVIDIA A100 gpus. The distributed training was performed with Pytorch Lightning and DeepSpeed stage 2 [6] with full precision. We set the initial learning rate to 1e-5, with 10000 warmup steps. We used AdamW [14] optimizer with a learning

[5] https://github.com/aub-mind/arabert/

[6] https://www.deepspeed.ai/tutorials/zero/

rate linear decay, β_1 set to 0.9, β_2 set to 0.98, ϵ set to 1e-5, weight decay set to 0.2. We used a maximum norm of the gradients set to 0.1. We train keeping the maximum sequence length to 256. As mentioned in the paper, we train 4-layer Multilingual BERT (mBERT) continually pretrained for 800K steps and 4-layer BERT model trained from scratch with C1 for 1.3M steps. We find that especially for 4-layer mBERT training beyond that with the corpus we had and the hyper-parameters leads to overfitting as observed from the validation loss. We use the same MLM masked loss strategy of 15% masking but only setting 90% of these 15% to the MASK token and the rest 10% to a random token.

A.3 Finetuning details

We use the open-source code for the ALUE benchmark [7] and all the default hyper-parameters set here. We set the maximum sequence length here to 256 and we train the model for 10 epochs.

A.3.1 Evaluation on ALUE Data

Task Type	Task Name	Domain
Single Sentence Classification	MDD	Travel
	OOLD	Tweet
	OHSD	Tweet
	FID	Tweet
Sentence Pair Classification	MQ2Q	Web
	XLNI	Misc
Multi-label classification	SEC	Tweet
Regression	SVREG	Tweet

Table 4: The task categories in ALUE benchmark

1. IDAT@FIRE2019 Irony Detection Task (FID) : The shared task of Irony Detection in Arabic Tweets is based on a dataset of around 5,000 tweets. Each tweet is labeled with a "1" when it is ironic, holds satire, parody, sarcasm, or if the intended meaning is the contrary of the literal one. A label of "0" is given otherwise.
2. MADAR Shared Task Subtask 1 (MDD): Each sentence is exclusively classified into one of 25 labels, corresponding to one city out of 25 predefined Arab cities.
3. NSURL-2019 Shared Task 8 (MQ2Q): In this task, a pair of questions is assumed to be semantically similar if they have the same exact answer and meaning, which is denoted with a label of "1". A label of "0" is given otherwise.
4. OSACT4 Shared Task on Offensive Language Detection (OOLD & OHSD): For both of these tasks, the data contains 32,000 comments. We refine this corpus with 8 multi-labeled fine-grained classes, namely: toxic, insult, threat, identity hate, sexual, racial, blasphemy, and politically incorrect. Each label of these is denoted with either "1" if the class applies, or "0" otherwise.
5. SemEval-2018 Task 1 - Affect in Tweets (SVREG & SEC): The first is the Emotion Classification task (SEC) in which a tweet is classified using one or more of eleven possible labels that best capture the emotions expressed by it. These labels are anger, anticipation, disgust, fear, joy, love, optimism, pessimism, sadness, surprise, and trust. The second is the Sentiment Intensity Regression task (SVREG) in which participants are expected to predict the "valence" of a given tweet, using a real-valued score between "0" and "1", with "0" representing the most negative sentiment possible, while "1" being the most positive sentiment possible.

[7] https://github.com/Alue-Benchmark/alue_baselines/blob/master/bert-baselines/run_alue.py

6. Cross-lingual Sentence Representations (XNLI): These sentence pairs are labeled with either one of the following logical relationship labels: entailment, contradictory, or neutral. The data was originally labeled in the English language and then translated into 15 other languages including Arabic.