Deep Active Learning for Morphophonological Processing

Seyed Morteza Mirbostani, Yasaman Boreshban
Dpt. of Computer Engineering, University of Guilan, Rasht, Iran
{m.mirbostani, boreshban}@msc.guilan.ac.ir

Salam Khalifa, Seyed Abolghasem Mirroshandel, and **Owen Rambow**
Dpt. of Linguistics and Institute for Advanced Computational Science (IACS)
Stony Brook University, Stony Brook, USA
{first.last}@stonybrook.edu

Abstract

Building a system for morphological processing is a challenging task in morphologically complex languages like Arabic. Although there are some deep learning based models that achieve successful results, these models rely on a large amount of annotated data. Building such datasets, specially for some of the lower-resource Arabic dialects, is very difficult, time-consuming, and expensive. In addition, some parts of the annotated data do not contain useful information for training machine learning models. Active learning strategies allow the learner algorithm to select the most informative samples for annotation. There has been little research that focuses on applying active learning for morphological inflection and morphophonological processing. In this paper, we have proposed a deep active learning method for this task. Our experiments on Egyptian Arabic show that with only about 30% of annotated data, we achieve the same results as does the state-of-the-art model on the whole dataset.

1 Introduction

Recently, there has been lots of interest in morphological (re-)inflection processing (Narasimhan et al., 2015; Kirov and Cotterell, 2018; Belth et al., 2021). Having an acceptable model for morphological processing will help improve the performance of different natural language processing (NLP) tasks like speech synthesis (Halabi, 2016), morphological disambiguation (Khalifa et al., 2020; Inoue et al., 2021), and machine translation (Sennrich and Haddow, 2016; Erdmann et al., 2019; Alhafni et al., 2020). Despite recent progress in this field of study, there are lots of challenges for low-resource languages. Especially for utilizing successful but data-hungry deep learning (DL) models, the need for annotated data is vital. However, data annotation is a hard, expensive, and time-consuming task. In addition, lots of annotated data does not contain useful information for improving the quality of a learning algorithm. In this paper, we propose a deep active learning (DAL) algorithm for morphophonological processing that is able to decrease the need for annotated data, by using only informative samples. In our experiments, we have chosen Arabic, a morphologically rich language. In addition, lots of Arabic dialects are very low-resource and the results from this study can help in building required datasets in a smarter way. Among Arabic dialects, Cairene Egyptian Arabic has been selected, because it is well-studied and has many resources and it is appropriate for our DAL simulation experiments. It should be noted that the proposed method is not specific to Arabic and it can be utilized on other languages or dialects.

As our baseline, we have chosen a very successful transformer model for character-level transduction tasks (Wu et al., 2021). We propose a pool-based DAL method in this study. To find the most uncertain (informative) samples, we combine an entropy strategy with a clustering method to keep an acceptable balance between the uncertainty and diversity of the chosen samples. The results of our experiments on the selected dataset show the success of the proposed DAL method.

2 Previous Work

In this section, we give a brief review of morphological inflection processing methods. Some recent DAL methods will also be reviewed.

There are several approaches for applying DL models for the morphological inflection problem (Yang et al., 2022; Wehrli et al., 2022; Batsuren et al., 2022; Wu et al., 2021; Dankers et al., 2021), which achieve successful results on different languages. Most of these models use character-level neural transducers using transformers, data augmentation, and recurrent

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 793–803

neural network (RNN)s.

In Arabic, there is also much non-neural research on morphological modeling, including finite state technology (Habash and Rambow, 2006), precompiled tabular morphological analyzers (Buckwalter, 2002, 2004; Graff et al., 2009; Taji et al., 2018), and allomorphy modeling through linguistically descriptive rules (Habash et al., 2022).

In recent years, DAL has been used in some sub-fields of NLP. Zhang et al. (2017) and Ru et al. (2020) achieved satisfactory results in text classification using active learning (AL) with convolutional neural networks and adversarial uncertainty sampling, respectively. Different acquisition functions using the conditional random field model have been applied in named entity recognition (Shen et al., 2017; Prabhu et al., 2019; Liu et al., 2020). In neural machine translation, Peris and Casacuberta (2018) used an attention-based function, and Liu et al. (2018) applied a reinforcement learning method. In another study, Zhao et al. (2020) proposed a word frequency-based acquisition function to train neural machine translation actively. More recently, Muradoglu and Hulden (2022) introduced a DAL method for lemma inflection in written modern standard Arabic and some other languages. Their method uses entropy at the word level, while we find that the max of character-level entropy for a word performs best.

3 Background and Problem Definition

Morphophonology involves studying the relationship between morphology and phonology. The goal is to analyze data in order to discover the underlying forms and ordered rules that explain the observed data (Hayes, 2008). Arabic morphophonology is particularly interesting due to its complex templatic and concatenative morphology. Changes in morphophonology can happen on the stem pattern and also on word boundaries and stem. In addition, phonological alterations can be triggered by the addition of morphemes in concatenative morphology cases. The main problem of this paper is morphophonological generation, in which an underlying representation (UR) is transformed into a surface form (SF). We also investigate analysis, i.e., learning to transform a SF to a UR.

UR	SF	SF Arabic
#$Ayil=In=uh#	#$aylInu#	شايلينه
#HAfiZ=In=hA#	#HafZinha#	حافظينها
#bi-ti-SAdf=U#	#bitSadfu#	بتصادفوا

Table 1: Samples of (UR, SF) pairs of the used dataset.

	TRAIN	DEV	EVAL
All	13,170	5,180	6,974
OOV	-	2,189	2,271

Table 2: The sizes of different splits of the used dataset.

4 Dataset

We use the Arabic morphophonology dataset created by Khalifa et al. (2022). It uses a broad phonemic transcription. They generated URs from the CALIMA$_{EGY}$ morphological analyzer (Habash et al., 2012) for every SF extracted from the ECAL dataset (Kilany et al., 2002). They also added the analyzer's segmentation to the UR part, delimiting word boundaries with #, prefixes with −, and suffixes with =. The dataset contains pairs of (UR, SF) as shown in Table 1. The split of this dataset is based on ECAL's split, which contains TRAIN, DEV, and EVAL subsets. Due to the fact that ECAL's splits are based on running texts, some words can occur in more than one split. Therefore Khalifa et al. (2022) also created subsets of the DEV and EVAL sets, called DEV-OOV and EVAL-OOV, which only contain non-overlapping words with the TRAIN split. The sizes of these splits are given in Table 2.

5 Proposed Method

In this section, we give a brief description of the baseline network. Then, the proposed DAL method will be explained in more detail.

5.1 Baseline Network

We have done several experiments to choose the most successful model for our AL experiments. Among different existing successful approaches (e.g., (Wehrli et al., 2022) and (Wu et al., 2021)), we chose Wu et al. (2021)'s system as our baseline system for conducting the DAL experiments because of its successful results on the utilized dataset. This is a transformer-based model that outperformed existing sequence-to-sequence models based on RNN for character-level transduction tasks, such as

morphological inflection generation, and achieved state-of-the-art results. Due to the fact that in character-level transduction tasks, the dataset sizes are significantly smaller than other tasks like machine translation, they have proposed a smaller transformer. In the next subsection, we will describe our proposed algorithm.

5.2 Active Learning Method

In this research, we have used the pool-based AL method in combination with the entropy strategy and clustering to determine uncertain samples based on the model's prediction preserving the diversity of chosen samples. The AL method considers all the available data of the pool, $\mathcal{U}$, which contains 13,170 samples, unannotated. Initially, about 10% of the samples (i.e., 1,400 samples) are chosen randomly from $\mathcal{U}$ for the data annotation process. In the first cycle of training the model, 500 samples from these 1,400 annotated samples are used for tuning, $\mathcal{T}$, and the rest of the labeled samples (i.e., 900 samples) are used as the initial training dataset, $\mathcal{L}$. The tune dataset is fixed throughout the procedure and determines the model with the highest accuracy during each AL training cycle. However, the training dataset, $\mathcal{L}$, is increased by δ samples (i.e., 250 samples) per training cycle.

After training the model on the $\mathcal{L}$ dataset for the first time, all the pool samples are passed to the model for prediction. For each UR, the probability values are determined by computing the softmax of the model's output $logits$. Most data sampling strategies in the AL method are based on some uncertainty criteria. In the case of sequence-to-sequence models focusing on character-level tasks, the sampling method based on entropy criteria is a suitable choice for uncertainty detection.

The output $logits$ for each character of the predicted SF word, w_{SF}, corresponds to the elements of a character vocabulary generated according to the predicted SF words. Using Equation (1), each set of $logits$, $\mathbf{ch}$, is used to calculate the probability vector of a character in w_{SF}. Here, $P_i(\mathbf{ch})$ is the probability value of the i^{th} element in the probability vector, ch_i is the $logit$ of the i^{th} element, and N is the vocabulary size.

$$P_i(\mathbf{ch}) = \frac{e^{ch_i}}{\sum_{i=1}^{N} e^{ch_i}} \tag{1}$$

The entropy of a character $\mathbf{ch}$, $E'(\mathbf{ch})$, is calculated by Equation (2) based on the probability values of all possible generated characters vocabulary.

$$E'(\mathbf{ch}) = -\sum_{i=1}^{N} P_i(\mathbf{ch}) \log P_i(\mathbf{ch}) \tag{2}$$

Equation (3) determines the entropy of the word w_{SF} by choosing the maximum value among all its characters' entropy values. That is, predicted labels with the lowest confidence have the highest entropy.

$$E(w_{\text{SF}}) = \max_{\mathbf{ch} \in w_{\text{SF}}} E'(\mathbf{ch}) \tag{3}$$

In each AL cycle, the trained model selects the next cycle's additional (informative) samples. δ number of UR words with the highest w_{SF} entropy are sampled without replacement. According to Equation (4), these most informative samples, w^*, are annotated and combined with the current $\mathcal{L}$ dataset to be utilized by the baseline system in the next AL cycle for training.

$$\mathbf{w}^* = \arg\max_{\mathbf{w} \in \mathcal{U}} E(\mathbf{w}) \tag{4}$$

We augment this approach with a clustering technique to maintain diversity during the sampling process. In this approach, α number of UR words, $\alpha > \delta$, with the highest w_{SF} entropy are selected for clustering. The optimal number of clusters is determined by computing the sum of the squared error (SSE) for various cluster counts, k. In each observation, S_i, centroids are determined by taking the mean value, μ_i, of all the points, w, in each cluster. The sum of the deviation of the points from centroids for all clusters combined determines SSE. An observation's cluster count, k, is optimal only if its SSE is **minimal** among all observations.

$$\text{SSE} = \sum_{i=1}^{k} \sum_{w \in S_i} ||w - \mu_i||^2 \tag{5}$$

The best cluster count in each AL cycle is used for clustering. The ultimate goal is to find δ most informative UR words with an acceptable diversity of total α (i.e., 1,000) samples.

A character-based word embedding model based on RNN is trained on datasets $\mathcal{L}$ and $\mathcal{T}$ to extract features for performing efficient clustering. Using a word embedding model enables us to have a sense

of UR words and their semantic relations with their corresponding SF words, considering $\mathcal{L}$ and $\mathcal{T}$ data points are labeled.

A word embedding model is used to vectorize α samples selected from $\mathcal{U}$. Two sequences of one-hot vectors, representing a pair of UR and SF, pass through two long short-term memory (LSTM) networks, and the norm of the difference between output embeddings is fed to a sigmoid activation function. A sample from the dataset is a related pair, and samples with the highest Levenshtein distance from each other are non-related. These samples are combined and used to train the network. The model converts a word to a vector based on a character vocabulary generated from the unique characters of the training set.

After standardizing the vectors, they are fed to principal component analysis (PCA) in order to retain the most significant features and minimize the computational costs by reducing dimensions. The features dimension is reduced to 3 components for clustering. α samples were divided into clusters using the k-means method. Then, proportional to the size of clusters, δ samples are selected from each group. These samples will be annotated and moved to $\mathcal{L}$ for the next AL training cycle.

6 Experimental Results

The details of the model's parameters and other variables in our experiments can be seen in Appendix A. To evaluate our proposed DAL method, we have run all experiments 5 times and the average and standard deviation of accuracy are visualized in Figure 1. We have reported random training (passive learning), AL with entropy, and AL with combined entropy and clustering method for each cycle of training.

As can be seen in Figure 1, the proposed DAL method (with and without clustering) grows much faster than the random curve and presents an asymptotical shape which shows that it has extracted all the useful information present in $\mathcal{U}$ when it reaches the asymptote using only 4,000 samples (i.e., about 30% of the training set). In contrast, the random (passive) learner requires the entire training set to achieve maximum performance. This is true for all evaluation sets, except for EVAL-OOV, where the random (passive) learner reaches maximum performance after 6,000 samples. We have no explanation for this unusual behavior of EVAL-OOV, but we do observe that the overall top performance on EVAL-OOV is 2 percentage points lower than on DEV-OOV, while EVAL and DEV have similar results. This supports the hypothesis that EVAL-OOV contains some very difficult cases which the underlying system (with active or passive learning) cannot learn, so that active and passive learning converge earlier.

Unfortunately, we could not obtain evidence from this dataset that the proposed hybrid method (combining entropy and clustering) is effective.

Our model is designed to be language and model independent. It can be applied on different languages and it can be used on top of different character-based DL models. However, the proposed method should be applied on new languages and dialects in further studies. We expect that further studies will show that clustering can be effective, especially if many diverse phenomena are present in the data.

7 Error Analysis

We performed an error analysis on 100 randomly chosen errors from the DEV-OOV set, generated by models trained on 1,900 samples using AL with entropy and random selection methods. On this sample size, the random selection approach achieves an error rate of 12.9%, which reduces to 7.8% through AL (a 40% error reduction). We distinguish three basic types of errors: the system deletes a letter (i.e., phoneme) it should not; the system changes a letter it should not; the system adds a letter it should not, or does not delete a letter it should. We summarize results in Table 3. Starting at the top, we see that letter deletion is greatly reduced by moving from random selection to AL; affix deletion is a special case, where an entire affix is not realized, and this problem is eliminated. Letter change is also greatly reduced. However, a special case of letter change becomes more prominent: vowel length changes. This is a common effect in Arabic phonology due to changes in syllabification resulting from adding affixes. Finally, we see that letter addition remains a problem, with the special case of the system failing to delete a letter in fact increasing. The only case which is reduced is the special case of i-deletion in the active participle, which the AL setting appears to learn much better. We then have a fairly small category with multiple errors, which remains about the same. As we expect when the error rate goes down, the proportion of problem cases in the

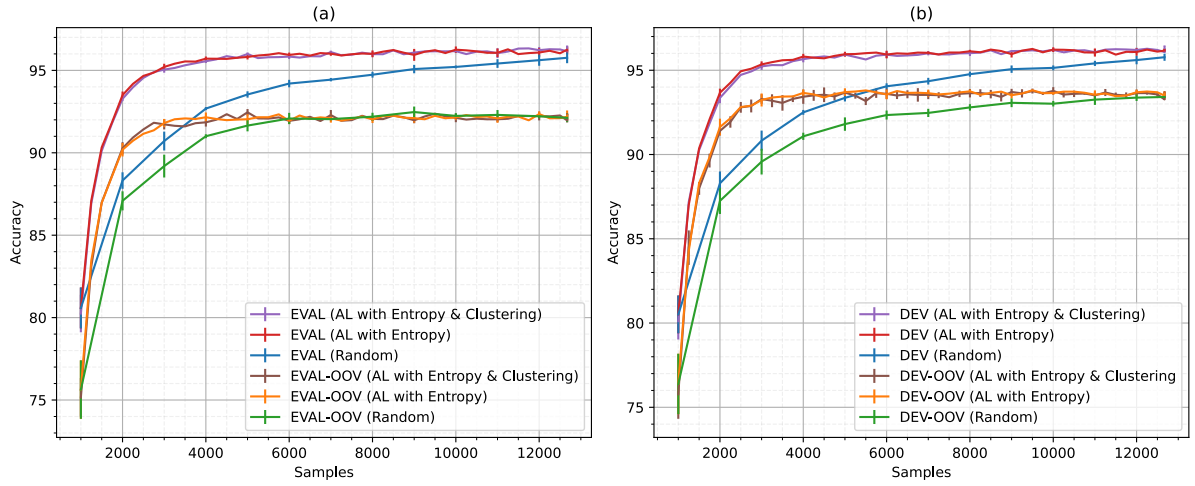

Figure 1: Inflection task (UR to SF): The average and standard deviation of the accuracy of the baseline model trained using AL with entropy, AL with combined entropy and clustering, and random training methods, and evaluated on (a) EVAL, EVAL-OOV, (b) DEV, and DEV-OOV datasets

corpus goes up. We distinguish three cases. First, foreign words have lexically idiosyncratic rules which cannot be learned. Second, almost all Arabic dialects replace the /l/ of the definite determiner /Al/ with the first letter of the following noun if it is coronal ("sun-letter assimilation"). However, Egyptian optionally also does this for /j/ and /k/, and the optionality is reflected in the training data which makes consistent learning impossible. Third, the corpus has a number of actual errors in the gold standard, usually in the UR. So in summary, the AL system has improved in all error types except for the letter addition categories.

Error Type	AL	Random
letter deletion	11	25
affix deletion	0	2
letter change	11	17
no v shortening	5	2
v shortening	7	4
letter addition	8	6
no letter deletion	6	2
no AP i deletion	2	5
multiple errors	14	13
foreign	1	1
sun letter	10	8
gold error	25	15
Sum	100	100

Table 3: Error analysis (counts)

8 Conclusion

In this paper, we have proposed a deep active learning method for the morphological inflection processing task. The proposed method can be used on different languages; however, as a case study, we have focused on the Egyptian Arabic dialect. The results of our experiment demonstrate the outstanding efficiency of the proposed method: With only 30% of the total training dataset, we achieve the same accuracy as the state-of-the-art model trained on whole dataset.

Future research includes applying this method to different low-resources Arabic dialects and other languages for building datasets, using other baseline algorithms, working on new uncertainty measures, and exploring for which datasets the clustering method can be helpful. We also intend to investigate how we can exploit our insight from the error analysis that the letter addition cases remain high (or even increase).

Acknowledgements

We would like to thank three anonymous reviewers for their comments. Experiments were performed on the SeaWulf HPC cluster maintained by RCC and the Institute for Advanced Computational Science (IACS) at Stony Brook University and made possible by National Science Foundation (NSF) grant No. 1531492.

Limitations

Like lots of deep learning algorithms, our work also needs GPU resources. In common learning problems, models will be trained once on the existing training datasets, using dev sets for tuning the models. Then the trained model would be ready for use. In contrast, in active learning, we need to train the model several times (i.e., whenever new annotated samples are added to the current training set, the model should be re-trained), which increases the need for GPU resources. However, the need for GPU, is not related to our proposed method and it is due to the nature of active learning. In addition, one can run the active learning method once (rather than iteratively) for building an acceptable dataset.

It should be noted that we have designed the algorithm in a way to be independent of the target language and utilized model. However, we only tested our method on Egyptian Arabic dialect and the accuracy of the model should be investigated on other languages and dialects using different learning models in further studies.

Ethics Statement

The current work is a fundamental research and it is not essentially related to a particular application. We do not predict any ethical concerns from the algorithms and technologies proposed in this work. We have utilized publicly available dataset and open source libraries, which have been published before.

References

Bashar Alhafni, Nizar Habash, and Houda Bouamor. 2020. Gender-aware reinflection using linguistically enhanced neural models. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 139–150.

Khuyagbaatar Batsuren, Gábor Bella, Aryaman Arora, Viktor Martinovic, Kyle Gorman, Zdeněk Žabokrtský, Amarsanaa Ganbold, Šárka Dohnalová, Magda Ševčíková, Kateřina Pelegrinová, Fausto Giunchiglia, Ryan Cotterell, and Ekaterina Vylomova. 2022. The SIGMORPHON 2022 shared task on morpheme segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 103–116, Seattle, Washington. Association for Computational Linguistics.

Caleb Belth, Sarah Payne, Deniz Beser, Jordan Kodner, and Charles Yang. 2021. The greedy and recursive search for morphological productivity. *arXiv preprint arXiv:2105.05790*.

Tim Buckwalter. 2002. Buckwalter Arabic morphological analyzer version 1.0. Linguistic Data Consortium (LDC) catalog number LDC2002L49, ISBN 1-58563-257-0.

Tim Buckwalter. 2004. Buckwalter Arabic Morphological Analyzer Version 2.0. LDC catalog number LDC2004L02, ISBN 1-58563-324-0.

Verna Dankers, Anna Langedijk, Kate McCurdy, Adina Williams, and Dieuwke Hupkes. 2021. Generalising to German plural noun classes, from the perspective of a recurrent neural network. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, pages 94–108.

Alexander Erdmann, Salam Khalifa, Mai Oudah, Nizar Habash, and Houda Bouamor. 2019. A little linguistics goes a long way: Unsupervised segmentation with limited language specific guidance. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 113–124.

David Graff, Mohamed Maamouri, Basma Bouziri, Sondos Krouna, Seth Kulick, and Tim Buckwalter. 2009. Standard Arabic Morphological Analyzer (SAMA) Version 3.1. Linguistic Data Consortium LDC2009E73.

Nizar Habash, Ramy Eskander, and Abdelati Hawwari. 2012. A morphological analyzer for egyptian Arabic. In *Proceedings of the twelfth meeting of the special interest group on computational morphology and phonology*, pages 1–9.

Nizar Habash, Reham Marzouk, Christian Khairallah, and Salam Khalifa. 2022. Morphotactic modeling in an open-source multi-dialectal Arabic morphological analyzer and generator. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 92–102, Seattle, Washington. Association for Computational Linguistics.

Nizar Habash and Owen Rambow. 2006. MAGEAD: A morphological analyzer and generator for the Arabic dialects. In *Proceedings of the International Conference on Computational Linguistics and the Conference of the Association for Computational Linguistics (COLING-ACL)*, pages 681–688, Sydney, Australia.

Nawar Halabi. 2016. *Modern standard Arabic phonetics for speech synthesis*. Ph.D. thesis, University of Southampton.

Bruce Hayes. 2008. *Introductory phonology*, volume 7. John Wiley & Sons.

Go Inoue, Salam Khalifa, and Nizar Habash. 2021. Morphosyntactic tagging with pre-trained language models for Arabic and its dialects. *arXiv preprint arXiv:2110.06852*.

Salam Khalifa, Jordan Kodner, and Owen Rambow. 2022. Towards learning Arabic morphophonology. In *Proceedings of the seventh Arabic Natural*

Language Processing Workshop (WANLP) at EMNLP 2022, pages 295–301s.

Salam Khalifa, Nasser Zalmout, and Nizar Habash. 2020. Morphological analysis and disambiguation for Gulf Arabic: The interplay between resources and methods. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3895–3904.

H Kilany, H Gadalla, H Arram, A Yacoub, A El-Habashi, and C McLemore. 2002. Egyptian colloquial Arabic lexicon. *LDC catalog number LDC99L22*.

Christo Kirov and Ryan Cotterell. 2018. Recurrent neural networks in linguistic theory: Revisiting pinker and prince (1988) and the past tense debate. *Transactions of the Association for Computational Linguistics*, 6:651–665.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning to actively learn neural machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 334–344.

Mingyi Liu, Zhiying Tu, Zhongjie Wang, and Xiaofei Xu. 2020. Ltp: A new active learning strategy for bert-crf based named entity recognition. *ArXiv*, abs/2001.02524.

Saliha Muradoglu and Mans Hulden. 2022. Eeny, meeny, miny, moe. how to choose data for morphological inflection. *arXiv preprint arXiv:2210.14465*.

Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. 2015. An unsupervised method for uncovering morphological chains. *Transactions of the Association for Computational Linguistics*, 3:157–167.

Álvaro Peris and Francisco Casacuberta. 2018. Active learning for interactive neural machine translation of data streams. *arXiv preprint arXiv:1807.11243*.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. *arXiv preprint arXiv:1909.09389*.

Dongyu Ru, Jiangtao Feng, Lin Qiu, Hao Zhou, Mingxuan Wang, Weinan Zhang, Yong Yu, and Lei Li. 2020. Active sentence learning by adversarial uncertainty sampling in discrete space. *arXiv preprint arXiv:2004.08046*.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. *arXiv preprint arXiv:1606.02892*.

Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*.

Dima Taji, Jamila El Gizuli, and Nizar Habash. 2018. An Arabic dependency treebank in the travel domain. In *Proceedings of the Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT)*, Miyazaki, Japan.

Silvan Wehrli, Simon Clematide, and Peter Makarov. 2022. Cluzh at sigmorphon 2022 shared tasks on morpheme segmentation and inflection generation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 212–219.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2021. Applying the transformer to character-level transduction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1901–1907, Online. Association for Computational Linguistics.

Changbing Yang, Garrett Nicolai, Miikka Silfverberg, et al. 2022. Generalizing morphological inflection systems to unseen lemmas. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 226–235.

Ye Zhang, Matthew Lease, and Byron Wallace. 2017. Active discriminative text representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Yuekai Zhao, Haoran Zhang, Shuchang Zhou, and Zhihua Zhang. 2020. Active learning approaches to enhancing neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1796–1806.

A Implementation Details

We used a character-level transducer based on transformers (Wu et al., 2021) as the baseline in our computational experiments. A transformer with 4 encoder-decoder layers, 4 self-attention heads, the embedding dimension of 256, and the hidden size of 1024 for the feed-forward layer is used. The number of parameters of the model with these specifications is 7.37M, excluding embeddings and the pre-softmax linear layer.

Experiments were performed on a system with an Intel Core i7-8700K CPU 3.70GHz 6-Core, a GeForce GTX 1080 8GB, and 64GB of memory. The minimum resources required for each AL training cycle is 3.38GBs of GPU and 3.5GBs of RAM. A training cycle completes in less than 60 minutes.

The best hyper-parameter values of the experiment are given in Table A.1. We conducted multiple experiments with different values. For AL cycle sampling, various methods such as maximum entropy, maximum entropy limited to vowel letters, and mean entropy were employed. However, the results of the maximum entropy outperformed others. Moreover, we performed all the experiments 5 times and reported the average

and standard deviation of the results on Figure 1.

Regarding the implementation of the proposed algorithm, we have used PyTorch, NumPy, Pandas, Matplotlib, scikit-learn, and chars2vec software packages.

Parameter	Value
AL Initial Sampling Method	random
AL Cycle Sampling Method	entropy
AL Cycle Clustering Method	k-means
AL Initial Training Samples Counts	900
AL Tuning Samples Counts	500
AL Pre-clustering Samples Counts	1000
AL Cycle Samples Counts	250
Training Batch Size	400
Evaluation Batch Size	16
Dropout	0.3
Character Embedding Dimension	50
PCA Components	3
Max Cluster Counts	8

Table A.1: The best hyper-parameter values of the experiment

B Additional Results

Employing the proposed method, we conducted the experiments in the reversed direction (i.e., SF to UR) for morphophonological analysis. As demonstrated in Figure 2, the DAL methods outperformed random training by extracting all the informative samples of the pool set, $\mathcal{U}$, when reaching the asymptote using 8,000 samples. Since the current baseline is designed for morphophonological generation tasks, its performance is diminished for SF to UR. As our proposed method is model-agnostic, a more suitable baseline model for this task would achieve higher accuracies in morphophonological analysis for both passive and active learning.

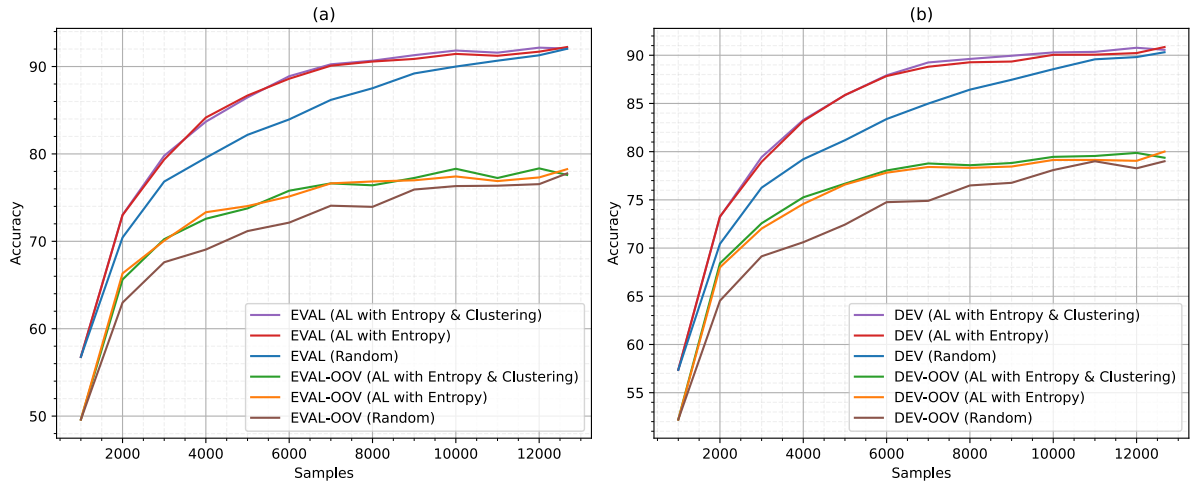

Figure 2: Analysis task (SF to UR): The average accuracy of the baseline model trained using AL with entropy, AL with combined entropy and clustering, and random training methods, and evaluated on (a) EVAL, EVAL-OOV, (b) DEV, and DEV-OOV datasets for morphological inflection analysis

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
In the Limitation section after Paper's conclusion.

☑ A2. Did you discuss any potential risks of your work?
In Ethics Statement section after Limitation section.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Please refer to the abstract and introduction (Section 1) of the paper.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Sections 3 and 4.

☑ B1. Did you cite the creators of artifacts you used?
Sections 3, 4, and appendix A.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Sections 3 and 4.

C ☑ Did you run computational experiments?

Appendix A.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5, Appendix A, and Appendix B.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.