FAMIE: A Fast Active Learning Framework for Multilingual Information Extraction

Minh Van Nguyen[1], **Nghia Trung Ngo**[1], **Bonan Min**[2], **and Thien Huu Nguyen**[1]
[1] Dept. of Computer and Information Science, University of Oregon, Eugene, OR, USA
[2] Raytheon BBN Technologies, USA
{minhnv@cs,nghian@,thien@cs}.uoregon.edu,
bonan.min@raytheon.com

Abstract

This paper presents FAMIE, a comprehensive and efficient active learning (AL) toolkit for multilingual information extraction. FAMIE is designed to address a fundamental problem in existing AL frameworks where annotators need to wait for a long time between annotation batches due to the time-consuming nature of model training and data selection at each AL iteration. This hinders the engagement, productivity, and efficiency of annotators. Based on the idea of using a small proxy network for fast data selection, we introduce a novel knowledge distillation mechanism to synchronize the proxy network with the main large model (i.e., BERT-based) to ensure the appropriateness of the selected annotation examples for the main model. Our AL framework can support multiple languages. The experiments demonstrate the advantages of FAMIE in terms of competitive performance and time efficiency for sequence labeling with AL. We publicly release our code (https://github.com/nlp-uoregon/famie) and demo website (http://nlp.uoregon.edu:9000/). A demo video for FAMIE is provided at: https://youtu.be/I2i8n_jAyrY.

1 Introduction

Information Extraction (IE) systems provide important tools to extract structured information from text (Li et al., 2014; Nguyen and Nguyen, 2019; Lai et al., 2021; Veyseh et al., 2021; Nguyen et al., 2021a). At the core of IE involves sequence labeling tasks that aim to recognize word spans and semantic types for some objects of interest (e.g., entities and events) in text. For example, two typical sequence labeling tasks in IE feature Named Entity Recognition (NER) to find names of entities of interest, and Event Detection (ED) to identify triggers of specified event types (Walker et al., 2006). Despite extensive research effort for sequence labeling (Lafferty et al., 2001; Ma and Hovy, 2016; Pouran Ben Veyseh et al., 2021b), a major bottleneck of existing IE methods involves the requirement for large-scale human-annotated data to build high-quality models. As annotating data is often expensive and time-consuming, large-scale labeled data is not practical for various domains and languages.

To address the annotation cost for IE, previous work has resorted to active learning (AL) approaches (Settles and Craven, 2008; Settles, 2009) where only a selective set of examples are annotated to minimize the annotation effort while maximizing the performance. Starting with a set of unlabeled data, AL methods train and improve a sequence labeling model via multiple human-model collaboration iterations. At each iteration, three major steps are performed in order: (i) training the model on the current labeled data, (ii) using the trained model to select the most informative examples in the current unlabeled set for annotation, and (iii) presenting the selected examples to human annotators to obtain labels. In AL, the number of annotated samples or annotation time might be limited by a budget to make it realistic.

Unfortunately, despite much potentials, existing AL methods and frameworks are still not applied widely in practice due to their main focus on devising the most effective example selection algorithm for human annotation, e.g., based on the diversity of the examples (Shen et al., 2017; Yuan et al., 2020) and/or the uncertainty of the models (Roth and Small, 2006; Wang and Shang, 2014; Shelmanov et al., 2021). Training and selection time in the first and second steps of each AL interaction is thus not considered in prior work for sequence labeling. This is a critical issue that limits the application of AL: annotators might need to wait for a long period between annotation batches due to the long training and selection time of the models at each AL iteration. Given the widespread trend of using large-scale pre-trained language models (e.g., BERT), this problem of long waiting or training/selection time in AL can only become worse. On the one hand, the long idle time of annotators reduces the number of annotated examples given an annotation budget. Further, the engagement of annotators in the annotation process can drop significantly due to the long interruptions between annotation rounds, potentially affecting the quality of their produced annotation. In all, current AL frameworks are unable to optimize the available time of annotators to maximize the annotation quantity and quality for satisfactory performance.

To this end, we demonstrate a novel AL framework (called FAMIE) that leverages large-scale pretrained language models for sequence labeling to achieve optimal modeling capacity while significantly reducing the waiting time between annotation rounds to optimize annotator time. Instead of training the full/main large-scale model for data selection at each AL iteration, our key idea is to train only a small proxy model on the current labeled data to recommend new examples for annotation in the next round. In this way, the training and data selection time can be reduced significantly to enhance annotation engagement and quality. An important issue in this idea is to ensure that the examples selected by the proxy model are also optimal for the main large model. To this end, we introduce a novel knowledge distillation mechanism for AL that encourages the synchronization between the proxy and main models, and promotes the fitness of selected examples for the main model. To update the main model with new annotated data for effective distillation, we propose to train the main large model on current labeled data during the annotation time, thus not adding to the waiting time of annotators between annotation rounds. This is in contrast to previous AL frameworks that leave the computing resources unused during annotation time. Our approach can thus efficiently exploit both human and computer time for AL.

To evaluate the proposed AL framework FAMIE, we conduct experiments for multilingual sequence labeling problems, covering two important IE tasks (i.e., NER and ED) in three languages (i.e., English, Spanish, and Chinese). The experiments demonstrate the efficiency and effectiveness of FAMIE that can achieve strong performance with significantly less human-computer collaboration time. Compared to existing AL systems such as ActiveAnno (Wiechmann et al., 2021) and Paladin (Nghiem et al., 2021), our system FAMIE features important advantages. First, FAMIE introduces a novel approach to reduce model training and data selection time for AL via a small proxy model and knowledge distillation while still benefiting from the advances in large-scale language models. Second, while previous AL systems only focus on some specific task in English, FAMIE can support different sequence labeling tasks in multiple languages due to the integration of our prior multilingual toolkit Trankit (Nguyen et al., 2021b) to perform fundamental NLP tasks in 56 languages. Third, in contrast to previous AL systems that only implement one data selection algorithm, FAMIE covers a diverse set of AL algorithms. Finally, FAMIE is the first complete AL system that allows users to define their sequence labeling problems, work with the models to annotate data, and eventually obtain a ready-to-use model for deployment.

2 System Description

In AL, we are given two initial datasets, a small seed set of labeled examples $D_0 = \{(\mathbf{w}, \mathbf{y})\}$ and an unlabeled example set $U_0 = \{\mathbf{w}\}$ (the seed set D_0 is optional and our system can work directly with only U_0). For sequence labeling, models consume a sequence of K words $\mathbf{w} = [w_1, w_2, \ldots, w_K]$ (i.e., a sentence/example) to output a tag sequence $\mathbf{y} = [y_1, y_2, \ldots, y_K]$ (y_i is the label tag for w_i). The tag sequence is represented in the BIO scheme to capture spans and types of objects of interest.

A typical AL process contains multiple rounds/iterations of model training, data selection, and human annotation in a sequential manner. Let D and U be the overall labeled and unlabeled set of examples at the beginning of the current t-th iteration (initialized with D_0 and U_0). At the current iteration, a sequence labeling model is first trained on the current labeled set D. A sample selection algorithm then employs the trained model to suggest the most informative subset of examples U^t in U (i.e., $U^t \subset U$) for annotation. Afterwards, a human annotator will provide labels for the sentences in the selected set U^t, leading to the labeled examples D^t for U^t. The labeled and unlabeled sets can then be updated via: $D \leftarrow D \cup D^t$ and $U \leftarrow U \setminus U^t$.

2.1 Model

We employ the typical Transformer-CRF architecture for sequence labeling (Nguyen et al., 2021b). In particular, given the input sentence

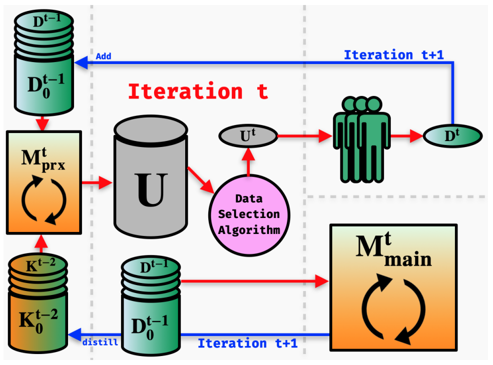


Figure 1: The overall Proxy Active Learning process.

$\mathbf{w} = [w_1, w_2, \dots, w_K]$, the state-of-the-art multilingual language model XLM-Roberta (Conneau et al., 2020) is used to obtain contextualized embeddings for the words: $\mathbf{X} = \mathbf{x}_1, \dots, \mathbf{x}_K = \text{XLMR}(w_1, \dots, w_K)$ (i.e., to support multiple languages). Afterwards, the word embeddings are sent to a feed-forward network with softmax in the end to obtain the score vectors: $\mathbf{z}_i = \text{softmax}(\mathbf{h}_i)$ where $\mathbf{h}_i = \text{FFN}(\mathbf{x}_i)$. Here, each value in $\mathbf{z}_i$ represents a score for a tag in the tag set V. The score vectors are then fed into a Conditional Random Field (CRF) layer to compute a distribution for possible tag sequences for $\mathbf{w}$: $P(\hat{\mathbf{y}}|\mathbf{w}) = \frac{\exp(s(\hat{\mathbf{y}},\mathbf{w}))}{\sum_{\hat{\mathbf{y}}' \in Y(\mathbf{w})} \exp(s(\hat{\mathbf{y}}',\mathbf{w}))}$ where $Y(\mathbf{w})$ is the set of all possible tag sequences for $\mathbf{w}$. Also, $s(\hat{\mathbf{y}}, \mathbf{w})$ is the score for a tag sequence $\hat{\mathbf{y}} = [\hat{y}_1, \dots, \hat{y}_K]$: $s(\hat{\mathbf{y}}, \mathbf{w}) = \sum_i \mathbf{z}_i[\hat{y}_i] + \sum_i \pi_{\hat{y}_i \to \hat{y}_{i+1}}$. Here, $\pi_{\hat{y}_i \to \hat{y}_{i+1}}$ is the transition score from the tag $\hat{y}_i$ to the tag $\hat{y}_{i+1}$. The model is trained by minimizing the negative log likelihood: $L_{task} = -\log P(\mathbf{y}|\mathbf{w})$. For inference, the Viterbi algorithm is used for decoding: $\hat{\mathbf{y}}^* = \max_{\hat{\mathbf{y}}'} P(\hat{\mathbf{y}}'|\mathbf{w})$.

Adapter-based Finetuning To further improve the memory and time efficiency, we incorporate light-weight adapter networks (Houlsby et al., 2019; Peters et al., 2019) into our model. In form of small feed-forward networks, adapters are injected in between the transformer layers of XLM-Roberta. For training, we only update the adapters while the parameters of XLM-Roberta are fixed. This significantly reduces the amount of learning parameters while sacrificing minimal extraction loss, or in case of low-resource learning even surpassing performance of fully fine-tuned models.

2.2 Data Selection Strategies

To improve the flexibility to accommodate different problems, our AL framework supports a wide range of data selection strategies for choosing the best batch of examples to label at each iteration for sequence labeling. These algorithms are categorized into three groups, i.e., uncertainty-based, diversity-based, and hybrid. For each group, we explore its most popular methods as follows.

Uncertainty-based. These methods select examples for annotation according to the main model's confidence over the predicted tag sequences for unlabeled examples. Early methods sort the unlabeled examples by the uncertainty of the main model. To avoid the preference over longer examples, the method Maximum Normalized Log-Probability (**MNLP**) (Shen et al., 2017) proposes to normalize the likelihood over example lengths. In particular, MNLP selects examples with the highest MNLP scores: $MNLP(\mathbf{w}) = -\max_{\hat{\mathbf{y}}'} \frac{1}{K} \log P(\hat{\mathbf{y}}'|\mathbf{w})$.

Diversity-based. Algorithms in this category assume that a representative set of examples can act as a good surrogate for the whole dataset. **BERT-KM** (Yuan et al., 2020) uses K-Means to cluster the examples in unlabeled data based on the contextualized embeddings of the sentences (i.e., the representations for the [CLS] tokens in the trained BERT-based models). The nearest neighbors to the K cluster centers are then chosen for labeling.

Hybrid. Recently, several works have proposed data selection strategies for BERT-based AL to balance between uncertainty and diversity. The **BADGE** method (Ash et al., 2019; Kim, 2020) chooses examples from clusters of gradient embeddings, which are formed with the token representations $\mathbf{h}_i$ from the penultimate layer of the main model and the gradients of the cross-entropy loss with respect to such token representations. The gradient embeddings are then sent to the K-Means++ to find the initial K cluster centers that are distant from each other, serving as the selected examples (Kim, 2020).

In addition, we implement the AL framework **ALPS** (Yuan et al., 2020) that does not require training the main model for data section. ALPS employs the surprisal embedding of $\mathbf{w}$, which is obtained from the likelihoods of masked tokens from pre-trained language models (i.e., XLM-Roberta). The surprisal embeddings are also clustered to select annotation examples as in BERT-KM.

2.3 Proxy Active Learning

As discussed in the introduction, model training and data selection at each iteration of traditional AL methods might consume significant time (especially with the current trend of large-scale language models), thus introducing a long idle time for annotators that might reduce annotation quality and quantity. To this end, (Shelmanov et al., 2021) have explored approaches to accelerate training and data selection steps for AL by leveraging smaller and approximate models during the AL iterations. To make it more efficient, the main large model is only trained once in the end over all the annotated examples in AL. Unfortunately, this approach suffers from the mismatch between the approximate and main models as they are separately trained in AL, thus limiting the effectiveness of the selected examples for the main model (Lowell et al., 2019).

To overcome these issues, our AL framework FAMIE trains a small proxy network at each iteration to suggest new unlabeled samples. Dealing with the mismatch between the proxy-selected examples and the main model, FAMIE proposes to involve the main model in the training and data selection for the proxy model. In particular, at each AL iteration, the main model will still be trained over the latest labeled data. However, to avoid the interference of the main large model with the waiting time of annotators, we propose to train the main model during the annotation time of the annotators (i.e., main model training and data annotation are done in parallel). Given the main model trained at previous iteration, knowledge distillation will be employed to synchronize the knowledge between the main and proxy models at the current iteration.

The complete framework for FAMIE is presented in Figure 1. At iteration t, a proxy acquisition model is trained on the current labeled data set $D_0^{t-1} = D^0 \cup D^1 \ldots \cup D^{t-1}$. The trained proxy model at the current step is called M_{prx}^t. Also, we use knowledge distillation signals K_0^{t-2} that is computed from the main model M_{main}^{t-1} trained at the previous iteration $t-1$ to synchronize the proxy model M_{prx}^t and the main model M_{main}^{t-1} (M_{prx}^1 is trained only on D^0). Afterwards, a data selection algorithm is used to select a batch of examples U^t from the current unlabeled set U for annotation, leveraging the feedback from M_{prx}^t. Next, a human annotator will label U^t to produce the labeled data batch D^t for the next iteration $t+1$. During this annotation time, the main model will also be trained again over the current labeled data D_0^{t-1} to produce the current version M_{main}^t of the model. The distillation signal K_0^{t-1} for the next step will also be computed after the training of M_{main}^t. This process is repeated over multiple iterations and the last version of M_{main} will be returned for users.

To improve the fitness of the proxy-based selected examples for M_{main}, we leverage the distilled version miniLM of XLM-Roberta (Wang et al., 2021) that employs similar stacks of transformer layers for the proxy model M_{prx}. Note that M_{prx} also includes a CRF layer on top of miniLM.

2.4 Uncertainty Distillation

Although the proxy and main model M_{prx} and M_{main} are trained on similar data, they might still exhibit large mismatch, e.g., regarding decision boundaries. This prompts a demand for regularizing the proxy model's predictions to be consistent with those of a trained main model to improve the fitness of the selected examples for M_{main}. Ideally, we expect the tag sequence distribution $P_{prx}(\mathbf{y}|\mathbf{w})$ learned by the proxy model to mimic the tag sequence distribution $P_{main}(\mathbf{y}|\mathbf{w})$ learned by the main model. To this end, we propose to minimize the difference between the intermediate outcomes (i.e., the unary and transition scores) of the two distributions. In particular, we introduce the following distillation objective for each sentence $\mathbf{w}$ at one AL iteration: $L_{dist} = -\sum_i \sum_v p_i^{main}[v] \log p_i^{prx}[v] + \sum_i (\pi_{y_i \to y_{i+1}}^{main} - \pi_{y_i \to y_{i+1}}^{prx})^2$ where p_i^{main} and p_i^{prx} are the tag distributions computed by the main and proxy models respectively for the word $w_i \in \mathbf{w}$ (i.e., the scores $\mathbf{z}_i$). Note that p_i^{main} and $\pi_{y_i \to y_{i+1}}^{main}$ serve as the knowledge distillation signal that is obtained once the main model finishes its training at each iteration. Here, we will use the newly selected examples for the current annotation to compute the distillation signals. The overall objective to train M_{prx} at each AL iteration is thus: $L = L_{task} + L_{dist}$.

3 Usage

Detailed documentation for FaMIE is provided at: `https://famie.readthedocs.io/`. The codebase is written in Python and Javascript, which can be easily installed through PyPI at : `https://pypi.org/project/famie/`.

Initialization. To initialize a project, users first choose a data selection strategy and upload a label set to define a sequence labeling problem. Next,

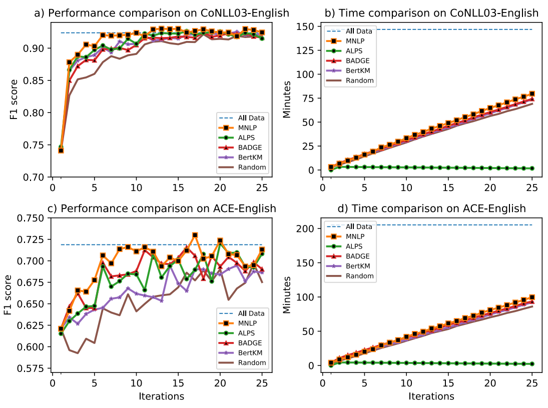
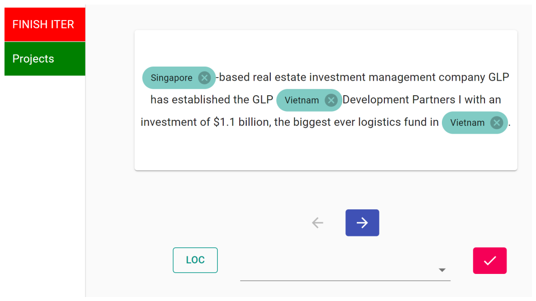


Figure 2: Comparison among data selection strategies.

the dataset U with unlabeled sentences should be submitted. FAMIE then allows users to interact with the models and annotate data over multiple rounds with a web interface. Also, FAMIE can detect languages automatically for further processing.

Annotating procedure. Given one annotation batch in an iteration, annotators label one sentence at a time as illustrated in Figure 3. In particular, the annotators annotate the word spans for each label by first choosing the label and then highlighting the appropriate spans. Also, FAMIE designs the size of the annotation batches to allow enough time to finish the training of the main model during the annotation time at each iteration.

Output. Unlike previous AL toolkits which focus only on their web interfaces to produce labeled data, FAMIE provides a simple and intuitive code interface for interacting with the resulting labeled dataset and trained main models after the AL processes. The code snippet in Figure 4 presents a minimal usage of our **famie** Python package to use the trained main model for inference over new data. This allows users to immediately evaluate their models and annotation efforts on data of interest.



Singapore -based real estate investment management company GLP has established the GLP Vietnam Development Partners I with an investment of $1.1 billion, the biggest ever logistics fund in Vietnam .

Figure 3: Annotation interface in FAMIE.

4 Evaluation

Datasets and Hyper-parameters. To comprehensively evaluate our AL framework FAMIE, we conduct experiments on two IE tasks (i.e., NER and ED) for three languages using four datasets: CoNLL03-English (Tjong Kim Sang and De Meulder, 2003) and CoNLL02-Spanish (Tjong Kim Sang, 2002) for NER, and ACE-English and ACE-Chinese for ED (i.e., using the multilingual ACE-05 dataset (Walker et al., 2006; Nguyen and Grishman, 2015, 2018)). The CoNLL datasets cover 4 entity types while 33 event types are annotated in ACE-05 datasets. We follow the standard data splits for train/dev/test portions for each dataset (Li et al., 2013; Lai et al., 2020; Pouran Ben Veyseh et al., 2021a).

For the main target model M_{main}, the full-scale XLM-Roberta$_{large}$ model is used as the encoder. Our framework for AL thus inherits the ability of XLM-Roberta to support more than 100 languages. Also, we employ the compact miniLM architecture (distilled from the pre-trained XLM-Roberta) for the proxy model M_{prx}. In all experiments, the main model is trained for 40 epochs while the proxy model is trained for 20 epochs at each iteration. We use the Adam optimizer with batch size of 16 and learning rate of $1e$-5 to train the models.

We follow the AL settings in previous work to achieve consistent evaluation (Kim, 2020; Shelmanov et al., 2021; Liu et al., 2022). Specifically, the unlabeled pool is created by discarding labels from the original training data of each dataset; 2% of which ($\sim$ 242 sentences) is selected for labeling at each iteration for a total of 25 iterations (examples of the first iteration are randomly sampled to serve as the seed D_0). The annotation is simulated by recovering the ground-truth labels of the corresponding instances. The model performance is measured on the test datasets by taking average over 3 runs with different random seeds.

```python
import famie
# access a project via its name
p = famie.get_project('NewProject')
# access the project's labeled data
data = p.get_labeled_data()

# access the project's trained target model
model = p.get_trained_model()
# make predictions with the trained model
doc = '''Nick is happy.'''
output = model.predict(doc)
print(output)
# [('Nick', 'B-Person'), ('is', 'O'), ('happy', 'O'), ('. ', 'O')]
```

Figure 4: Accessing the labeled dataset and the trained main model returned by an AL project.

Comparing Data Selection Strategies. In this experiment, we aim to determine the best data selection strategy for our AL framework. To this end, we perform the standard AL process (i.e., training the full transformer-CRF model with no adapters,

	Idle	CoNLL03-English						CoNLL02-Spanish						ACE-English						ACE-Chinese					
	mins/iter	10%	20%	30%	40%	50%	100%	10%	20%	30%	40%	50%	100%	10%	20%	30%	40%	50%	100%	10%	20%	30%	40%	50%	100%
Full Data	x	x	x	x	x	x	92.4	x	x	x	x	x	89.6	x	x	x	x	x	71.9	x	x	x	x	x	69.1
Large	41.6	**90.3**	**92.4**	**93.0**	**92.4**	92.4	x	86.9	**88.6**	**89.4**	**89.3**	89.0	x	**67.8**	**71.1**	**70.0**	**72.4**	**71.3**	x	**64.8**	67.6	**71.3**	68.7	**71.5**	x
FaMIE	**3.4**	90.1	91.7	91.8	91.7	**92.7**	x	86.5	88.2	88.5	88.1	**89.4**	x	67.0	69.3	69.5	68.9	70.6	x	61.3	**67.9**	68.5	**69.8**	69.6	x
FaMIE-A	5.7	89.7	90.8	91.3	91.9	91.7	x	**87.4**	87.2	89.0	87.7	89.1	x	67.2	68.0	69.5	68.9	70.6	x	62.8	66.5	67.9	66.3	69.4	x
FaMIE-AD	5.6	87.0	90.1	90.5	90.7	90.5	x	85.5	86.9	87.7	88.8	88.6	x	64.9	65.4	67.7	66.8	69.1	x	58.1	65.4	66.5	64.8	70.3	x
Random	x	86.0	89.1	90.6	91.4	91.9	x	80.8	85.3	88.1	88.7	88.6	x	60.4	64.1	66.9	69.0	67.5	x	48.4	58.2	65.1	65.4	66.6	x

Table 1: Main model's performance on multilingual NER and ED tasks. "Idle" indicate average waiting time of annotators.

selecting data, and annotating data at each iteration) for different data selection strategies to measure performance and time. We focus on English datasets in this experiment. Figure 2 reports the performance across AL iterations of the model for different data selection methods. As can be seen, "*MNLP*" is the overall best method for data selection in AL. We will thus leverage MNLP as the data section strategy for the evaluation of FAMIE.

Also, Figure 2 shows the annotators' idle time (the combined time for model training and data selection) across iterations for each selection strategy. The major difference comes from ALPS that has significantly less waiting time than other methods as it does not require model training. However, ALPS's performance is considerably worse than MNLP as a result, especially in early iterations. This demonstrates the importance of training and including the main model during the AL iterations for data section. Importantly, we find that the waiting time of annotators at each iteration is very high in current AL methods (e.g., more than 30 minutes after the first 8 iterations with the MNLP strategy), thus affecting the annotators' productivity.

Performance and Time Efficiency. To evaluate the performance and time efficiency of FAMIE, Table 1 compares our full proposed framework FAMIE (with proxy model, knowledge distillation, and adapters) with the following baselines: (i) "**Large**": the best AL baseline from the previous experiment employing the full-scale transformer encoder and MNLP for data selection; (ii) "**Random**": this is the same as "**Large**", but replaces MNLP with random selection; (iii) "**FAMIE-A**": this is the proposed framework FAMIE without adapter-based tuning (all parameters from the main model are fine-tuned); and (iv) "**FAMIE-AD**": we further remove the knowledge distillation loss from "**FAMIE-A**" in this method. The experiments are done for all four datasets of NER and ED.

The first observation is that FAMIE's performance is only marginally below that of Large despite only using the small proxy network for data selection. Importantly, annotators only have to wait for about 3.4 minutes per AL iteration before they can annotate the next data batch in FAMIE. This is over 10 times faster compared to the standard AL approaches (e.g., in Large). Second, the adapters in FAMIE not only boost the overall performance for AL but also reduce the waiting time for annotators. Also, we note that using adapters, the training time of M_{main} only takes 32 minutes at each iteration (on average). This is reasonable to fit into the time that an annotator needs to spend to label an annotation batch at each AL iteration, thus accommodating our proposal for training the main model during annotation time. Finally, FAMIE-AD performs worst (i.e., similar or even worse than Random) in most cases, which confirms the necessity of our distillation component in FAMIE.

5 Related Work

Despite the potential of AL in reducing annotation cost for a target task, most previous AL work focuses on developing data selection strategies to maximize the model performance (Wang and Shang, 2014; Sener and Savarese, 2017; Ash et al., 2019; Kim, 2020; Liu et al., 2022; Margatina et al., 2021). As such, previous AL methods and frameworks tend to ignore the necessary time to train models and perform data selection at each AL iteration that can be significantly long and hinder annotators' productivity and model performance. To make AL frameworks practical, few recent works have attempted to minimize the model training and data selection time by leveraging simple and non state-of-the-art architectures as the main model, e.g., ActiveAnno (Wiechmann et al., 2021) and Paladin (Nghiem et al., 2021). However, an issue with these approaches is the inability to exploit recent advances in large-scale language models to achieve optimal performance. In addition, some recent works have also explored large-scale language models for AL (Shelmanov et al., 2021; Yuan et al., 2020); however, to reduce waiting time for annotators, such methods need to exclude the training of the large models in the AL iterations or employ small models for data selection, thus suffering from a harmful mismatch between the annotated examples and the main models (Lowell et al., 2019).

6 Conclusion

We introduce FAMIE, a comprehensive AL framework that supports model creation and data annotation for sequence labeling in multiple languages. FAMIE optimizes the annotators' time by leveraging a small proxy network for data selection and a novel knowledge distillation to synchronize the proxy and main target models for AL. As FAMIE is task-agnostic, we plan to extend FAMIE to cover other NLP tasks in future work.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IUCRC Center for Big Learning. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2019. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *Proceedings of the International Conference on Machine Learning*.

Yekyung Kim. 2020. Deep active learning for sequence labeling based on diversity and uncertainty in gradient. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 1–8, Suzhou, China. Association for Computational Linguistics.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Viet Dac Lai, Minh Van Nguyen, Thien Huu Nguyen, and Franck Dernoncourt. 2021. Graph learning regularization and transfer learning for few-shot event detection. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2172–2176.

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Qi Li, Heng Ji, Yu Hong, and Sujian Li. 2014. Constructing information networks using one single model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*.

Mingyi Liu, Zhiying Tu, Tong Zhang, Tonghua Su, Xiaofei Xu, and Zhongjie Wang. 2022. Ltp: A new active learning strategy for crf-based named entity recognition. *Neural Processing Letters*, pages 1–22.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 21–30, Hong Kong, China. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Minh-Quoc Nghiem, Paul Baylis, and Sophia Ananiadou. 2021. Paladin: an annotation tool based on active and proactive learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 238–243, Online. Association for Computational Linguistics.

Minh Van Nguyen, Viet Lai, and Thien Huu Nguyen. 2021a. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 27–38, Online. Association for Computational Linguistics.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021b. Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *The 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Matthew E Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *RepL4NLP@ACL*.

Amir Pouran Ben Veyseh, Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021a. Unleash GPT-2 power for event detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Amir Pouran Ben Veyseh, Minh Van Nguyen, Nghia Ngo Trung, Bonan Min, and Thien Huu Nguyen. 2021b. Modeling document-level context for event detection via important context selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Dan Roth and Kevin Small. 2006. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pages 413–424. Springer.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Burr Settles. 2009. Active learning literature survey.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, Honolulu, Hawaii. Association for Computational Linguistics.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and Bayesian uncertainty estimates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1698–1712, Online. Association for Computational Linguistics.

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*.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Amir Pouran Ben Veyseh, Minh Van Nguyen, Bonan Min, and Thien Huu Nguyen. 2021. Augmenting open-domain event detection with synthetic data from gpt-2. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 644–660. Springer.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Technical report, Linguistic Data Consortium*.

Dan Wang and Yi Shang. 2014. A new active labeling method for deep learning. In *2014 International joint conference on neural networks (IJCNN)*, pages 112–119. IEEE.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pre-trained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2140–2151, Online. Association for Computational Linguistics.

Max Wiechmann, Seid Muhie Yimam, and Chris Biemann. 2021. ActiveAnno: General-purpose document-level annotation tool with active learning integration. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Demonstrations*, pages 99–105, Online. Association for Computational Linguistics.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.