AlpacaTag: An Active Learning-based Crowd Annotation Framework for Sequence Tagging

Bill Yuchen Lin[†*] **Dong-Ho Lee**[†*] **Frank F. Xu**[‡] **Ouyu Lan**[†] and **Xiang Ren**[†]
{yuchen.lin,dongho.lee,olan,xiangren}@usc.edu, frankxu@cmu.edu
[†]Computer Science Department, University of Southern California
[‡]Language Technologies Institute, Carnegie Mellon University

Abstract

We introduce an open-source web-based data annotation framework (AlpacaTag) for sequence tagging tasks such as named-entity recognition (NER). The distinctive advantages of AlpacaTag are three-fold. **1) Active intelligent recommendation**: dynamically suggesting annotations and sampling the most informative unlabeled instances with a back-end active learned model; **2) Automatic crowd consolidation**: enhancing real-time inter-annotator agreement by merging inconsistent labels from multiple annotators; **3) Real-time model deployment**: users can deploy their models in downstream systems while new annotations are being made. AlpacaTag is a comprehensive solution for sequence labeling tasks, ranging from rapid tagging with recommendations powered by active learning and auto-consolidation of crowd annotations to real-time model deployment.

1 Introduction

Sequence tagging is a major type of tasks in natural language processing (NLP), including *named-entity recognition* (detecting and typing entity names), *keyword extraction* (e.g. extracting aspect terms in reviews or essential terms in queries), *chunking* (extracting phrases), and *word segmentation* (identifying word boundary in languages like Chinese). State-of-the-art supervised approaches to sequence tagging are highly dependent on numerous annotations. New annotations are usually necessary for a new domain or task, even though transfer learning techniques (Lin and Lu, 2018) can reduce the amount of them by reusing data of other related tasks. However, manually annotating sequences can be time-consuming, expensive, and thus hard to scale.

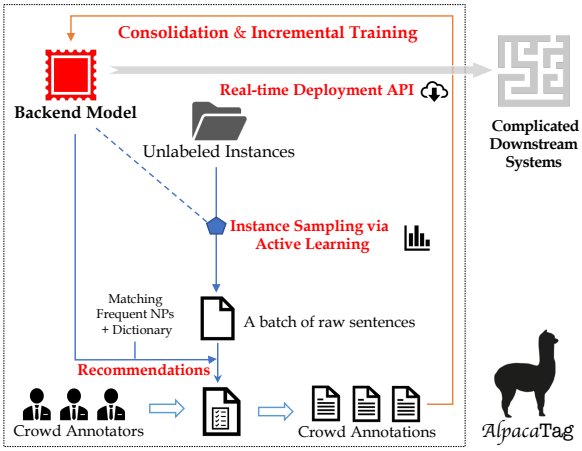

Figure 1: Overview of the AlpacaTag framework.

Therefore, it is still an important research question that how we can develop a better annotation framework to largely reduces human efforts.

Existing open-source sequence annotation tools (Stenetorp et al., 2012; Druskat et al., 2014a; de Castilho et al., 2016; Yang et al., 2018a) mainly focus on enhancing the friendliness of user interfaces (UI) such as data management, fast tagging with shortcut keys, supporting more platforms, and multi-annotator analysis. We argue that there are still three important yet underexplored directions of improvements: 1) active intelligent recommendation, 2) automatic crowd consolidation, 3) real-time model deployment. Therefore, we propose a novel web-based annotation tool named AlpacaTag[1] to address these three problems.

Active intelligent recommendation (§3) aims to reduce human efforts at both instance-level and corpus-level by learning a back-end sequence tagging model incrementally with incoming human annotations. At the instance-level, AlpacaTag applies the *model predictions* on current unlabeled sentences as tagging suggestions. Apart from that,

[*]Both authors contributed equally.

[1]The source code is publicly available at http://inklab.usc.edu/AlpacaTag/.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 58–63
Florence, Italy, July 28 - August 2, 2019.

we also greedily match *frequent noun phrases* and a dictionary of *already annotated spans* as recommendations. Annotators can easily confirm or decline such recommendations with our specialized UI. At the corpus-level, we use active learning algorithms (Settles, 2009) for selecting next batches of instances to annotate based on the model. The goal is to select the most *informative* instances for human to tag and thus to achieve a more cost-effective way of using human efforts.

Automatic crowd consolidation (§4) of the annotations from multiple annotators is an underexplored topic in developing crowd-sourcing annotation frameworks. As a crowdsourcing framework, `AlpacaTag` collects annotations from multiple (non-expert) contributors with lower cost and a higher speed. However, annotators usually have different confidences, preferences, and biases in annotating, which leads to possibly high inter-annotator disagreement. It is shown very challenging to train models with such noisy crowd annotations (Nguyen et al., 2017; Yang et al., 2018b). We argue that consolidating crowd labels during annotating can lead annotators to achieve real-time consensus, and thus decrease disagreement of annotations instead of exhausting post-processing.

Real-time model deployment (§5) is also a desired feature for users. We sometimes need to deploy a state-of-the-art sequence tagging model while the crowdsourcing is still ongoing, such that users can facilitate the developing of their tagging-required systems with our APIs.

To the best of our knowledge, there is no existing annotation framework enjoying such three features. `AlpacaTag` is the first unified framework to address these problems, while inheriting the advantages of existing tools. It thus provides a more comprehensive solution to crowdsourcing annotations for sequence tagging tasks.

In this paper, we first present the high-level structure and design of the proposed `AlpacaTag` framework. Three key features are then introduced in detail: *active intelligent recommendation* (§3), *automatic crowd consolidation* (§4), and *real-time model deployment* (§5). Experiments (§6) are conducted for showing the effectiveness of the proposed three features. Comparisons with related works are discussed in §7. Section §8 shows conclusion and future directions.

2 Overview of AlpacaTag

As shown in Figure 1, `AlpacaTag` has an actively learned back-end model (top-left) in addition to a front-end web-UI (bottom). Thus, we have two separate servers: a back-end **model server** and a front-end **annotation server**. The model server is built with `PyTorch` and supports a set of APIs for communications between the two servers and model deployment. The annotation server is built on `Django` in Python, which interacts with administrators and annotators.

To start annotating for a domain of interest, admins should login and create a new project. Then, they need to further import their raw corpus (with CSV or JSON format), and assign the tag space with associated *colors* and *shortcut keys*. Admins can further set the batch size to sample instances and to update back-end models. Annotators can login and annotate (actively sampled) unlabeled instances with tagging suggestions. We further present our three key features in the next sections.

3 Active Intelligent Recommendation

This section first introduces the back-end model (§3.1) and then presents how we use the back-end model for both instance-level recommendations (tagging suggestions, §3.2) as well as corpus-level recommendations (active sampling, §3.3).

3.1 Back-end Model: BLSTM-CRF

The core component of the proposed `AlpacaTag` framework is the back-end sequence tagging model, which is learned with an incremental active learning scheme. We use the state-of-the-art sequence tagging model as our back-end model (Lample et al., 2016; Lin et al., 2017; Liu et al., 2018), which is based on bidirectional LSTM networks with a CRF layer (BLSTM-CRF). It can capture character-level patterns, and encode token sequences with pre-trained word embeddings, as well as using CRF layers to capture structural dependencies in tagging. In this section, we assume the model is fixed as we are talking about how to use it for infer recommendations. How to update it by consolidating crowd annotations is illustrated in the Section §4.

3.2 Tagging Suggestions (Instance-level Rec.)

Apart from the inference results from the back-end model on the sentences, we also include two other kinds of tagging suggestions mentioned in

(a) the sentence and annotations are at the upper section; tagging suggestions are shown as underlined spans in the lower section.

(b) after click on a suggested span, a floating window will show up near for confirming the types (suggested type is bounded with red line).

(c) after click a suggested type or press a shortcut key (e.g. 'p'), confirmed annotations will show up in the upper annotation section.

Figure 2: The workflow for annotators to confirm a given tagging suggestion ("Hillary Clinton" as PER).

Fig. 1: *(frequent) noun phrases* and the dictionary of already tagged spans. Specifically, after admins upload raw corpora, AlpacaTag runs a phrase mining algorithm (Shang et al., 2018) to gather a list of frequent noun phrases, which are more likely to be entities. Admins can also optionally enable the framework to consider all noun phrases by chunking as span candidates. These suggestions are not typed. Additionally, we also maintain a dictionary mapping from already tagged spans in previous sentences to their majority tagged types. Frequent noun phrases and dictionary entries are matched by the greedy forward maximum matching algorithm. Therefore, the final tagging suggestions are merged from three sources with the following priority (ordered by their precision): dictionary matches > back-end model inferences > (frequent) noun phrases, while the coverage of the suggestions are in the opposite order.

Figure 2 illustrates how AlpacaTag presents the tagging suggestions for the sentence "Donald Trump had a dinner with Hillary Clinton in the white house.". The two suggestions are shown in the bottom section as underscored spans "Donald Trump" and "Hilary Clinton" (Fig. 2a). When annotators want to confirm "Hilary Clinton" as a true annotation, they first click the span and then click the right type in the appearing float window (Fig. 2b). They can also press the customized shortcut key for fast tagging. Note that the PER button is underscored in Fig. 2b, meaning that it is a recommended type for annotators to choose. Fig. 2c shows that after confirming, it is added into final annotations. We want to emphasize that our designed UI well solves the problem of confirming and declining suggestions. The float windows reduce mouse movement time very much and let annotators easily confirm or change types by clicking or pressing shortcuts to correct suggested types.

What if annotators want to tag spans not recommended? Normally annotating in AlpacaTag is as friendly as other tools: annotators can simply select the spans they want to tag in the upper section and click or press the associated type (Fig. 3). We implement this UI design based on an open-source Django framework named *doccano*[2].

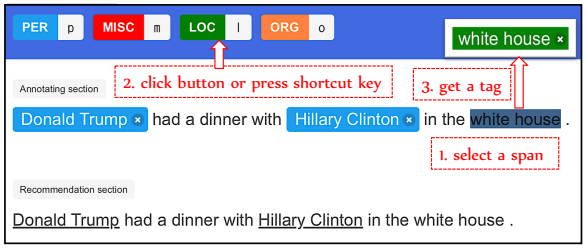

Figure 3: Annotating a span without recommendations.

3.3 Active Sampling (Corpus-level Rec.)

Tagging suggestions are at the instance level, while what instances we should ask annotators to label is also very important to save human efforts. Thus, the research question here is how to actively sample a set of the most *informative* instances from the whole unlabeled sentences, which is named active learning. The measure of informativeness is usually specific

[2] http://doccano.herokuapp.com/

to an existing model, which means what instances (if labeled correctly) can improve the model performance. A typical example of active learning is called *Least Confidence* (Culotta and McCallum, 2005), which applies the model on all the unlabeled sentences and treats the inference confidence as the negative of informativeness $(1 - \max_{y_1,\ldots,y_n} \mathbb{P}[y_1,\ldots,y_n|\{\mathbf{x}_{ij}\}])$. For `AlpacaTag`, we apply an improved version named Maximum Normalized Log-Probability (MNLP) (Shen et al., 2018), which eliminates the effect of length of sequences by averaging:

$$\max_{y_1,\ldots,y_n} \frac{1}{n}\sum_{i=1}^{n} \log \mathbb{P}[y_i|y_1,\ldots,y_{n-1},\{\mathbf{x}_{ij}\}]$$

Simply put, we manage to utilize the current back-end model for measuring the informativeness of unlabeled sentences and then sample the optimal next batch for annotators to label.

4 Automatic Crowd Consolidation

As a crowd-sourcing annotation framework, `AlpacaTag` aims to reduce inter-annotator disagreement while keeping their individual strengths in labeling specific kinds of instances. We achieve this by applying annotator-specific back-end models ("personal models") and consolidate them into a shared back-end model named "consensus model". Personal models are used for personalized active sampling, such that annotators will label the regions of data space that are most important to be labeled by them respectively.

Assume there are K annotators, we refer them as $\{A_1,\ldots,A_K\}$. For each annotator A_i, we refer the corresponding sentences and annotations processed by it to be $\mathbf{x}_i = \{x_{i,j}\}_{j=1}^{l_i}$ and $\mathbf{y}_i^{A_i} \in \mathcal{Y}^{l_i}$. The target of consolidation is to generate predictions for the test data $\mathbf{x}$, while the resulting labels are referred as $\hat{\mathbf{y}}$. Note that annotators may annotate different portions of the data and sentences can have inconsistent crowd labels.

As shown in Fig. 4, we model the behavior of every annotator by constructing annotator representation matrices $\mathbf{C}^k$ from the network parameters of *personal back-end models*. The personal models share the same BLSTM layers for encoding sentences and avoiding over-parameterizing. They are incrementally updated every batch, which usually consists of $50 \sim 100$ sentences sampled actively by the algorithms mentioned in §3.3. We further merge the crowd representation $\mathbf{C}^k$ into a representation matrix $\mathbf{C}$, and thus construct a consensus model by using $\mathbf{C}$ to generate tag scores in the BLSTM-CRF architecture, which is improved from the approach proposed by Nguyen et al. (2017).

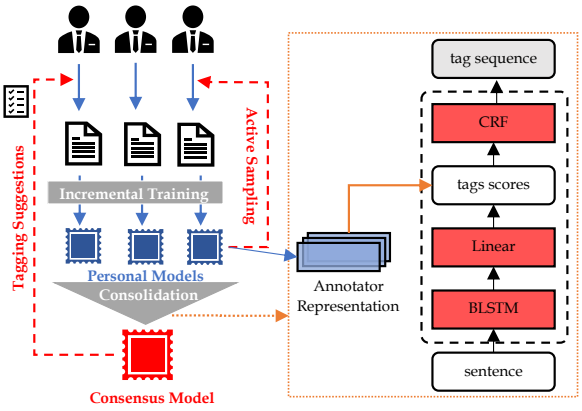

Figure 4: Reducing inter-annotator disagreement by training personal models for personalized active sampling and constructing consensus models.

In summary, `AlpacaTag` periodically updates the back-end consensus model by consolidating crowd annotations for annotated sentences through merging personal models. The updated back-end model continues to offer suggestions in future instances for all the annotators, and thus the consensus can be further solidified with recommendation confirmations.

5 Real-time Model Deployment

Waiting for a model to be trained with ready human annotations can be a huge bottleneck for developing complicated pipeline systems in information extraction. Instead, users may want a sequence tagging model that can be constantly updated, even when the annotation process is still undergoing. `AlpacaTag` naturally enjoys this feature since we maintain a back-end tagging model with incremental active learning techniques.

Thus, we provide a suite of APIs for interacting with the back-end consensus model server. They can work for both communication with the annotation server and real-time deployment of the back-end model. One can obtain inference results for their developing complicated systems by querying such APIs for downstream applications.

6 Experiments

To investigate the performance of our implemented back-end model with incremental active learning and consolidation, we conduct a prelim-

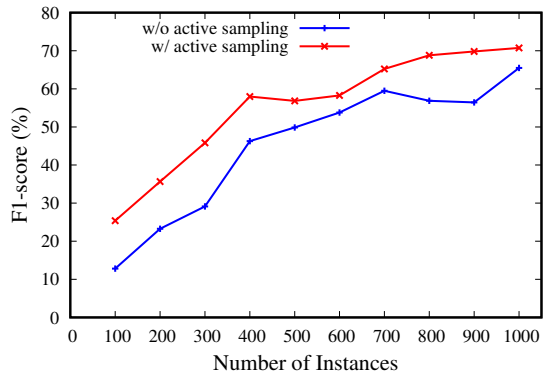

Figure 5: Effectiveness of incremental updating with and without active sampling.

inary experiment on the *CoNLL2003* dataset. We set iteration step size as 100 with the default order for single-user annotation, and incrementally update the back-end model for 10 iterations. Then, we apply our implementation for active sampling the optimal batches of samples to annotate dynamically. All results are tested on the official test set.

As shown in Fig. 5, the performance with active learning is indeed improved by a large margin over vanilla training effectively. We also find that with active sampling, using annotations of about 35% training data can achieve the performance of using 100%, confirming the reported results of previous studies (Siddhant and Lipton, 2018).

For consolidating multiple annotations, we test our implementation with the real-world crowdsourced data collected by Rodrigues et al. (2013) using Amazon Mechanical Turk (AMT). Our methods outperform existing approaches including MVT-SLM (Wang et al., 2018) and Crowd-X (Nguyen et al., 2017) (see Tab. 1).

7 Related Works

Many sequence annotation tools have been developed (Ogren, 2006; Bontcheva et al., 2013; Chen and Styler, 2013; Druskat et al., 2014b) for basic annotation features including data management, shortcut key annotation, and multi-annotation presentation and analysis. However, few of them enjoys the above-introduced three features of `AlpacaTag`. It is true that a few existing tools also support tagging suggestions (instance-level recommendations) as follows:

- `BRAT` (Stenetorp et al., 2012) and `GATE` (Bontcheva et al., 2013) can offer suggestions with a fixed tagging model like CoreNLP (Manning et al., 2014). However, it is hardly helpful when users need to annotate sentences with customized label set specific to their interested domains, which is the most common motivation for people to use an annotation framework.
- `YEDDA` (Yang et al., 2018a) simply generates suggestions by exact matching against a continuously updated lexicon of *already annotated spans*. In comparison, this is a subset of `AlpacaTag`'s recommendations. Note that `YEDDA` cannot suggest any unseen spans.
- `WebAnno` (Yimam et al., 2013) integrates a learning component for suggestions, which is based on hand-crafted features and generic online learning framework (MIRA). Nonetheless, the back-end model is too far away from the state-of-the-art methods for sequence tagging and hard to customize for consolidation and real-time deployment.

`AlpacaTag`'s tagging suggestions are more comprehensive since they are from state-of-the-art BLSTM-CRF architecture, annotation dictionary, and frequent noun phrase mining. An unique advantage is that it also supports active learning to sample instances that are most worth labeling to reduce human efforts. In addition, `AlpacaTag` further attempts to reduce inter-annotator disagreement during annotating by automatically consolidating personal models to consensus models, which pushes further than just presenting inter-annotator agreement without doing anything helpful.

While recent papers have shown the power of deep active learning with simulations (Siddhant and Lipton, 2018), a practical annotation framework with active learning is missing. `AlpacaTag` is not only an annotation framework but also a practical environment for evaluating different active learning methods.

Methods	Precision(%)	Recall(%)	F_1(%)
MVT-SLM	88.88(±0.25)	65.04(±0.80)	75.10(±0.44)
Crowd-Add	**89.74**(±0.10)	64.50(±1.48)	75.03(±1.02)
Crowd-Cat	89.72(±0.47)	63.55(±1.20)	74.39(±0.98)
Ours	88.77(±0.25)	**72.79**(±0.04)	**79.99**(±0.08)
Gold	92.12(±0.31)	91.73(±0.09)	91.92(±0.21)

Table 1: Comparisons of consolidation methods.

8 Conclusion and Future Directions

To sum up, we propose an open-source web-based annotation framework `AlpacaTag` that provides

users with more advanced features for reducing human efforts in annotating, while keeping the existing basic annotation functions like shortcut keys. Incremental active learning of back-end models with crowd consolidation facilities intelligent recommendations and reduce disagreement.

Future directions include supporting annotation of other tasks, such as relation extraction and event extraction. We also plan to design a real-time console for showing the status of back-end models based on *TensorBoard*, such that admins can better track the quality of deployed models and early stop annotating to save human efforts.

References

Kalina Bontcheva, Hamish Cunningham, Ian Roberts, Angus Roberts, Valentin Tablan, Niraj Aswani, and Genevieve Gorrell. 2013. Gate teamware: a web-based, collaborative text annotation framework. In *Proc. of LREC*.

Richard Eckart de Castilho, Éva Mújdricza-Maydt, Seid Muhie Yimam, Silvana Hartmann, Iryna Gurevych, Anette Frank, and Christian Biemann. 2016. A web-based tool for the integrated annotation of semantic and syntactic structures. In *Proc. of LT4DH@COLING*.

Wei-Te Chen and Will Styler. 2013. Anafora: a web-based general purpose annotation tool. In *Proc. of NAACL-HLT*.

Aron Culotta and Andrew McCallum. 2005. Confidence estimation for information extraction. In *Proc. of AAAI*.

Stephan Druskat, Lennart Bierkandt, Volker Gast, Christoph Rzymski, and Florian Zipser. 2014a. Atomic: an open-source software platform for multi-level corpus annotation. In *KONVENS*.

Stephan Druskat, Lennart Bierkandt, Volker Gast, Christoph Rzymski, and Florian Zipser. 2014b. Atomic: An open-source software platform for multi-level corpus annotation.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proc. of NAACL-HLT*.

Bill Y. Lin, Frank F. Xu, Zhiyi Luo, and Kenny Q. Zhu. 2017. Multi-channel bilstm-crf model for emerging named entity recognition in social media. In *NUT@EMNLP*.

Bill Yuchen Lin and Wei Lu. 2018. Neural adaptation layers for cross-domain named entity recognition. In *Proc. of EMNLP*.

Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Proc. of AAAI*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proc. of ACL*.

An Thanh Nguyen, Byron C. Wallace, Junyi Jessy Li, Ani Nenkova, and Matthew Lease. 2017. Aggregating and predicting sequence labels from crowd annotations. *Proc. of ACL*.

Philip V Ogren. 2006. Knowtator: a protégé plug-in for annotated corpus construction. In *Proc. of NAACL-HLT*.

Filipe Rodrigues, Francisco C. Pereira, and Bernardete Ribeiro. 2013. Sequence labeling with multiple annotators. *Machine Learning*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R. Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *Proc. of TKDE*.

Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *Proc. of ICLR*.

Aditya Siddhant and Zachary Chase Lipton. 2018. Deep bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proc. of EMNLP*.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. brat: a web-based tool for NLP-assisted text annotation. In *Proc. of EACL*.

Xuan Wang, Yu Zhang, Xiang Ren, Yuhao Zhang, Marinka Zitnik, Jingbo Shang, Curtis P. Langlotz, and Jiawei Han. 2018. Cross-type biomedical named entity recognition with deep multi-task learning. *Bioinformatics*.

Jie Yang, Yue Zhang, Linwei Li, and Xingxuan Li. 2018a. Yedda: A lightweight collaborative text span annotation tool. In *Proc. of ACL*.

YaoSheng Yang, Meishan Zhang, Wenliang Chen, Wei Zhang, Haofen Wang, and Min Zhang. 2018b. Adversarial learning for chinese ner from crowd annotations. In *Proc. of AAAI*.

Seid Muhie Yimam, Iryna Gurevych, Richard Eckart de Castilho, and Christian Biemann. 2013. Webanno: A flexible, web-based and visually supported system for distributed annotations. In *Proc. of ACL*.