LoReHLT19 System Description UMD-JHU

Mozhi Zhang, Jordan Boyd-Graber, Michelle Yuan,
C. Anton Rytting, Weiwei Yang, Philip Resnik
University of Maryland
College Park, MD

Ting Hua, Adam Poliak, Adam Teichert, Xu Han
Linghao Jin, João Sedoc, Benjamin Van Durme
Johns Hopkins University
Baltimore, MD

Abstract—**This document describes the JHU-UMD system for detecting situation frames for the 2019 LORELEI Evaluation in low resource languages. Four classification models predict SF types: Deep Averaging Networks (DAN), Classification Aided by Convergent Orthography (CACO), Adversarial Learning, and Learning to Rank. We also use an interface to refine cross-lingual word embeddings with native informant feedback.**

I. Introduction

For the SF task, we treat the problem as multilabel classification problem. We explored the following approaches to predict sf types: Deep Averaging Networks (DAN), Classification Aided by Convergent Orthography (CACO), Adversarial Learning, and Learning to Rank.

II. Submission Highlights

Compared to the evaluation of last year, we made these major changes:

- **Models**: We added two new models (Adversarial Learning and Learning to Rank).
- **NI session**: We use a new interface, CLIME, to interactively fine-tune cross-lingual word embeddings.

In addition to new models, we focused on a new parameter setting strategy based on the new SF evaluation paradigm. In particular, **we hypothesized that this year's focus on LOCATION should drive us to focus on recall in SF type prediction**, where for most hypothetical scenarios, we imagined that potential errors in SF type prediction would be random with respect to any given LOCATION. And therefore a higher recall SF type prediction would be more conducive to downstream *heatmap* inspired notions of finding the *hottest* locations.

III. Training Data

In this section, we will describe two datasets we collected to supplement the situation frame (SF) labels released in LoReLEI language packs: BBN annotation [11] and EASL annotation [10].

A. BBN annotation

For training data for the 2017 evaluation, we used annotations from a BBN-internal annotator on a set of English sentences from LORELEI language packs. As a sanity check and in order to determine a whitelist of mechanical turk annotators, we posted the sentences in this set to mechanical turk for annotation through our Computer Assisted Discovery Extraction and Translation [11, CADET] interface. By checking for suspicious distributions of labels and comparing turker labels to the official BBN annotations, we were able to determine which of the turkers were most likely to provide good annotations.

After this, Turkers annotate sentences from sets of incident-specific tweets collected from social media. We blacklisted anyone who had participated in the BBN data re-annotation yet wasnt included in the whitelist we had determined. This allowed us to get rid of untrustworthy Turkers while still allowing new Turkers to complete the assignment, which was necessary given the small size of the whitelist. For every twenty tweets that a Turker annotated, they would also annotate five random sentences from the BBN data that the official annotator had provided positive SF labels for. This allowed us to further refine our whitelist and helped us determine how much any new turkers were in agreement with the official BBN annotations. We collected annotations for several thousand English tweets in this manner, which we used as training data for an SF classifier. We used an active learning model to decide which tweets would be most useful to annotate, based on how likely each tweet was to confound our existing model.

We used these 2017 BBN labels and the turker labels on the same data to subselect inputs to annotate with scalars (as described below).

B. EASL annotation

In addition to binary annotations of SF types, we also collected **scalar** annotations (e.g., on a scale of 1–100, how much does this evoke a shelter need). We found that this prediction framework helps compensate for noisy annotations or inconsistent thresholds. This is further detailed in our ACL paper: Efficient Annotation of Scalar Labels (EASL) [10].

We apply these annotations to a large corpus of scalar labels on English data prior to the incident announcement. All labels were respect to situation frame types at that sentence level. Prior to the evaluation, we also obtained many crowd-sourced, sentence-level SF labels on language pack data, Doug Jones' tweets from an incident in Nepal [1], the reference English translations of the IL5 LoReLEI evaluation data, and eight additional tweet collections of various high-profile incidents, including:

- 2011 volcanic explosions in Eritrea
- 2011 major droughts in East Africa

- 2013 major 7.7-scale earthquake in Iran near Pakistan border
- 2013 overthrow of Morsi and replacement by el-Sissi in Egypthttps://www.overleaf.com/7694353544mbgyrmtzcpwq
- 2013 Cycone Phallin in India
- 2014 brutal crackdown on student protesters in Ethiopia
- 2011 mass flooding in Turkey
- 2015 ISIS suicide-attack & shooting in Paris

Our protocol (EASL) was to have workers focus on one SF type at a time and assign an integer value between 0 and 100 to each of five messages displayed in the same HIT on Amazon Mechanical Turk. Because we expected the data to be extremely sparse with respect to each type, we allocated resources by filtering the set in a SF type-specific way. In particular, for the language pack data and Doug Jones' tweets, we filtered most negative examples according to prior binary or ordinal annotation efforts. Likewise, for each SF type, we filter out all documents from the IL5 eval data that did not include a gold SF frame for that SF type. We used the resulting scalar annotations for language pack and Doug Jones' data to train an efficient leave-one-out cross-validated linear regressor from a dense sentence embedding extracted by a pre-trained, neural sentence encoder (Infersent). For each of the eight tweet groups, we first filtered based on time stamp so as to include only the hour of highest volume of tweets matching the respective keywords, the hour preceding the highest volume hour, and the 22 hours following it, giving a 24-hour window for each incident. Then for each SF type, we used scalar predictions from the trained regression model to score all tweets and then subsample from each of the eight incidents in a way that gave a roughly uniform distribution over scalar predictions.

IV. Embedding

For the models we describe in the next section, we use pretrained word cross-lingual embeddings as features. This allows us to project information across languages. Here we experimented with alignment methods for the word embeddings. There were three axes of variation, embedding dimension, alignment method, and bilingual lexical dictionary. The embeddings where assessed intrinsically using nearest-neighbor accuracy coming from "gold-standard" native informat annotations.

Our system starts with train word-level vectors trained using fastText skipgram [9]. All of our cross-lingual embeddings are aligned with the centered and normalized English and Bengali embeddings trained using Wikipedia data. The multilingual word vectors trained from LORELEI language packages. Subsequently, the IL embeddings are aligned using several methods: MCCA [1], VecMap [2], MUSE [5], and RCSLS [8] using vectors trained with the word-level skipgram fastText. Since these embeddings were in a shared space across languages, the model read in IL evaluation documents in the source language, using a classifier trained on the shared space. For all of the alignment methods we use a fully supervised setting, for alignment with English. As anchor (or lighthouse) words several dictionaries we have several possible sources (1) set0 category I dictionary, (2) NI annotations, (3) fast alignment (using fastalign), (4) machine translation bilingual lexicon, and (5) unimorph derived dictionary. As both (1) and (2) are "gold" annotations we always use these when available. As the number of correct anchor points improves alignment, all sources where tried, as well as all but MT, and all but MT and unimorph. Ultimately, our final cross-lingual system embeddings use MCCA and RCSLS with (1)+(2)+(3)+(5) and (1)+(3). After finding the best intrinsic performance of 100D word embeddings, we make 300D word embeddings since the intrinsic performance increased.

V. Situation Frame Prediction Models

We use four SF models: Deep Averaging Networks (DAN), Classification Aided by Convergent Orthography (CACO), Adversarial Learning, and Learning to Rank.

A. Deep Averaging Network

Our first model is a Deep Averaging Network [7, DAN]. DAN takes a sequence of cross-lingual word embeddings as input. To make a prediction, DAN passes the arithmetic mean of the input vectors through a multi-layer perceptron. The DAN is trained on our English training documents, using English word embeddings as input. At test time, we use two sets of features to perform cross-lingual transfer:

- IL embeddings (aligned with the English embeddings) of the original document.
- English embeddings of the MT outputs.

B. Classification Aided by Convergent Orthography

For Ilocano (IL12), we also submit a characteer-based system that transfers from the related language Tagalog. We use CACO [12], a framework that leverages subword similarities between related language pairs. CACO uses a character-level Bi-LSTM embedder to generate embeddings for each input word, and then passes these embeddings through a DAN classifier (Figure 1). CACO uses a joint character representation for both Tagalog and Ilocano and thus generalizes knowledge of Tagalog words to Ilocano words with similar forms.

The CACO needs to be trained on labeled Tagalog documents. Unfortunately, we did not annotate any Tagalog documents before the evaluation. As an approximation, we run an English SF model on the reference translations from the Tagalog language pack parallel texts and use the binarized model output as labels for Tagalog documents.

C. Adversarial Model

Our third modeling approach combines an adversarial classifier with a modern neural architecture in order to encourage our model to generalize across different disaster incidents. This is inspired by our recent work using adversarial learning to overcome dataset specific biases in Natural Language Inference [3], [4].

[1] Doug Jones, IL5 and Paris shooting data as well as some of the other language pack data were excluded from our submitted systems training sets in order to comply with the time-machine principle.

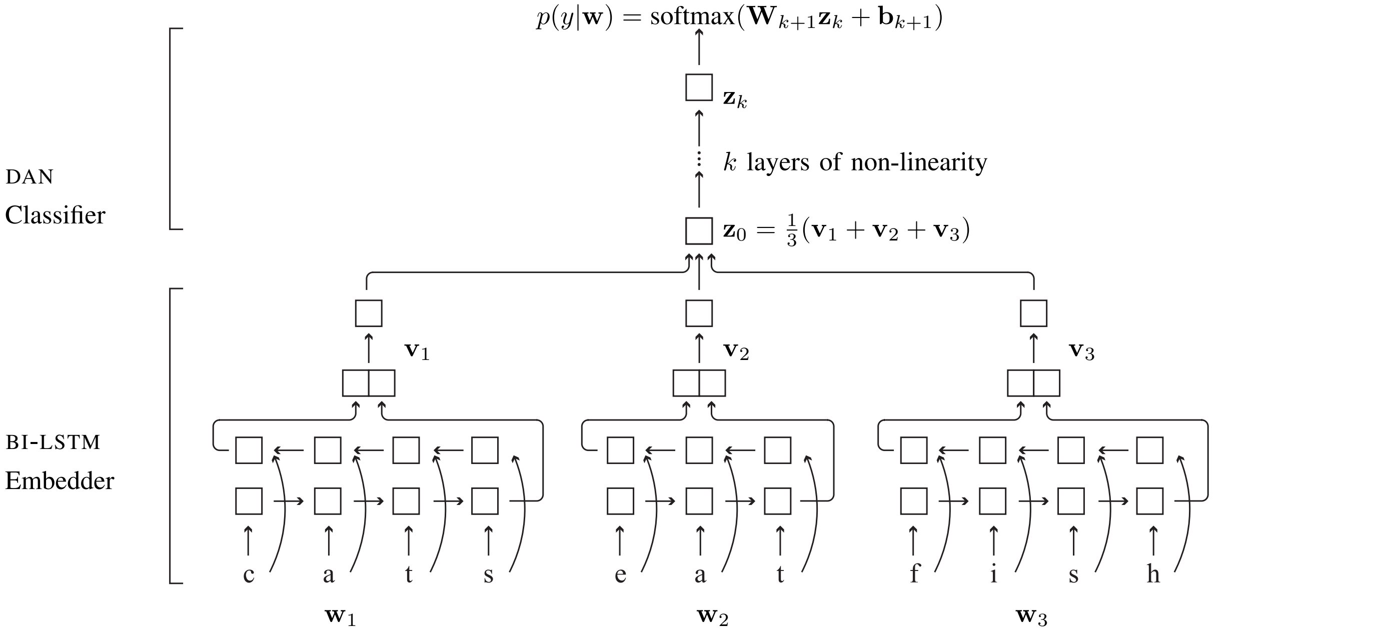

$p(y|\mathbf{w}) = \text{softmax}(\mathbf{W}_{k+1}\mathbf{z}_k + \mathbf{b}_{k+1})$
$\mathbf{z}_k$
k layers of non-linearity
$\mathbf{z}_0 = \frac{1}{3}(\mathbf{v}_1 + \mathbf{v}_2 + \mathbf{v}_3)$
DAN
Classifier
$\mathbf{v}_1$
$\mathbf{v}_2$
$\mathbf{v}_3$
BI-LSTM
Embedder
c a t s
e a t
f i s h
$\mathbf{w}_1$
$\mathbf{w}_2$
$\mathbf{w}_3$

Fig. 1: For each CACO frame, each input word $\mathbf{w}_i$ is mapped to a word vector $\mathbf{v}_i$ by passing its characters through a BI-LSTM embedder. The average of the word vectors $\mathbf{z}_0$ is then passed through k layers of non-linearity and a final softmax layer to predict the label y.

As our basline, we use a standard neural model. We encode our text with a Bi-LSTM encoder and pass the resulting text representation to a Multi-Layer Perceptron (MLP). We use a single model for each SF type, resulting in 11 independent neural models.

During training, we feed the representation from the Bi-LSTM to another MLP that predicts which incident (described in section III) the text originates from. However, we penalize our model for correctly predicting the source incident and reward it for incorrectly predicting the source incident. Following common practice in adversarial learning [6], we implement this penalization through a gradient reversal layer during back-propagation. We swept through model hyper-parameters that control the strength of the adversarial component.

In internal calibration tests using the annotations provided by the NI during the evaluation, an adversarial model outperformed the baseline network. We further used the NI's annotations to determine the best strength of the adversarial component.

D. Learning to Rank

Our training data based on EASL is gathered in batches of five: a single page has five distinct messages, each of which are assigned a scalar value from 1 'not present' to 100 'present'. Last year, and for most of our systems this year, we converted these values to binary judgments for sake of training binary classification models. This year we explored the idea that at model training time we should not be building binary classifiers, but instead learning to *rank* content as to how likely they evoke a given SF type. In addition, we wished to explore whether different crowdsource annotators may differ significantly in their annotations, or even whether a single annotator may make use of the scalar range a little differently from page to page (each batch of five). These ideas were explored by training models under a pair-wise loss: given two messages that were annotated under EASL, the learned representation of one message needed to have a derived score that was proportionate to the annotate scalar difference between that messagae and the paired message. E.g., a message saying *I'm hungry* would need to be *further away* from a message like *I'm bored*, then a message like *Food is good* (questionably evoking a FOOD need, but not confidently). After training models to rank content under each SF type, we calibrated our model scores against NI annotations to determine what the final binary thresholds should be. Internal experiments suggested that training based pairs of examples taken from single batches (pairs from the same page of five examples, by the same annotator) led to the most robust model. We have remaining experiments to perform to validate if this finding is robust, but our initial results were sufficiently promising to employ this approach for some of our evaluations submissions.

VI. Sentiment, Emotion, and Cognitive State

The Sentiment, Emotion, and Cognitive State (SEC) tasks focus on identifying subjective (i.e., sentiment-bearing or emotional) language, and, when found, inferring the polarity (positive or negative) and magnitude of sentiment, the presence or absence of three emotions (fear, anger, and joy), and the source of the sentiment (whether the author or some other entity).

The UMD SEC system employs a multi-stage support vector machine (SVM) model to identify subjective (non-neutral) segments within the provided LTF documents, and for each text segment deemed to contain subjective content, to further identify the segments sentiment polarity, emotion type, source, and (or the pilot) target. The multi-stage SVM model includes a series of binary SVM classifiers and SVRs. The classifiers and regressions take the text features (from LDC annotations and other sources) and source/target features as input and predict the texts emotion.

The classification hierarchy employed is as follows (after an initial binary SVM distinguishing subjective segments–i.e., those expressing positive or negative sentiment or emotion–from neutral ones):

- If "Subjective": Distinguish Polarity (Positive vs. Negative Sentiment)
  - Determine intensity, using Support Vector Regression (SVR)
  - If Positive: Distinguish Presence or Absence of Joy
  - If Negative: Distinguish Presence or Absence of Fear, Anger (two separate SVMs)
- If "Neutral": no SEC record generated for that segment.

The system was trained using 5-fold cross validation on the training data provided in the LDC training data for the 2018 pilot (LDC2018E79: LORELEI 2019 SEC Pilot Training Data V2). A version of the system was piloted using additional LDC-provided data from previous years, but the additional data hurt performance on the pilot evaluation, and so a system using only data from LDC2018E79 was fielded.

The SEC system developed for the pilot evaluation (and used for English-language documents for each incident) employed bag-of-words, part-of-speech, and lexical features, in addition to word embeddings. Since no training data was available for such features for the incident languages, the SEC system used for the incident languages IL11 and IL12 relied solely on word embeddings built as described in section IV and aligned to the English embeddings on which the system was trained.

In practice, we found that the system performed best in pilot data if augmented by several heuristics:

- Assume no positive sentiment or joy (i.e., set joy = False and polarity to negative)
- Assume that all social network (SN) documents have the *author* as source.
- Assume all news-wire (NW) documents have some *other* source (not from the entity KB).[1]

Since the final SF evaluation plan requires a single SEC prediction per source per situation frame, segment-level predictions were pooled to provide document-level predictions. Since preliminary analysis of the pilot evaluation data suggested that the system was less reliable in predicting SEC for shorter segments, we contemplated adding an additional heuristic that would exclude segments beneath a particular length threshold from consideration. However, this was not used in the final submissions.

VII. Native Informant Sessions

We use three tasks for our native informant (NI) sessions: word translation, IL document annotation, and CLIME.

a) Word Translation: For checkpoint 1, we ask NIs to translate a list of English words that are relevant to SF and SEC. The words are selected based on weights in a linear model. The collected translations are used to improve and validate cross-lingual word embeddings.

b) IL document annotation: During the evaluation, our NIs annotated two subsets of both set0 and set1 IL documents. Each pool of sftype-specific examples was constructed by using the model score of a preliminary system and subsampling 200 messages per sftype for each of set0 and set1 so as to approach a uniform distribution over model score for each SF type.

c) Refining embeddings with CLIME: We use CLIME, a new user interface that allows NI to interactively refine our cross-lingual word embeddings. Specifically, CLIME identifies a set of SF-relevant keywords and displays each keyword along with its nearest neighbors in the embedding space (Figure 2). The NI is asked to mark whether each neighbor induces the same labels as the keyword. CLIME then updates the embeddings based on NI feedbacks by pulling similar words closer and pushing dissimilar words away.

VIII. Output Integration

In this section, we will describe the integration process to produce the final submission outputs. Specifically, we first binarize text classifier results and then combine them with EDL locations from our collaborator in U Penn.

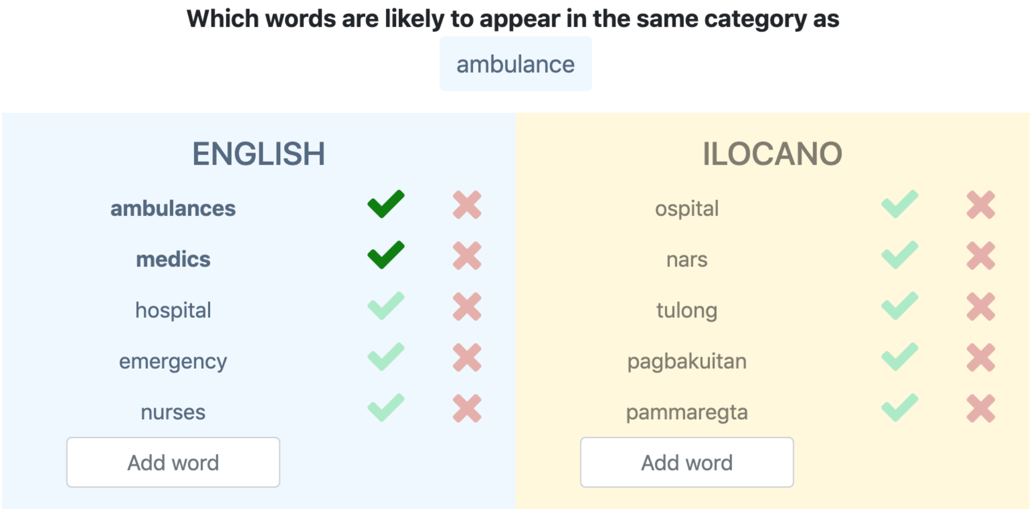
A. Model validation and binarization

The set of labeled IL sentences labeled by our NIs and team members became a validation set for choosing among candidate models and for calibrating (choosing thresholds for producing binary multi-label predictions based on scalar model scores). Given a set of candidate models with predictions on the validation set, we evaluated micro- and macro-averaged Spearman's rank correlation, and micro- and macro- averaged F1 scores based on three candidate fixed thresholds and a SF type-specific set of thresholds chosen to maximize F1 on the validation set as well as breakdown of rank correlation and F1 by SF type. For each IL, we determine ten submissions based on these results.

Given a base model (e.g. DAN), we considered submitting some combination of the following variants:

- **"Hard"** thresholds, where all SF type predictions are binarized at the same, fixed, threshold on model score; in some cases, we submitted multiple thresholds for the same base model;
- **"Opt"** threshold, where we used the threshold optimizing F1 on our validation set for that model and SF type,

[1] The heuristics for source were informed by discussion by other teams during the most recent Principal Investigators meeting.



Which words are likely to appear in the same category as

ambulance

ENGLISH			ILOCANO		
ambulances	✓	✗	ospital	✓	✗
medics	✓	✗	nars	✓	✗
hospital	✓	✗	tulong	✓	✗
emergency	✓	✗	pagbakuitan	✓	✗
nurses	✓	✗	pammaregta	✓	✗

Fig. 2: CLIME interface. A SF-relevant keyword is displayed at the top, and we ask NI to annotate a list of English and IL words that are closest to the keyword in the cross-lingual word embedding space.

optionally interpolated (we use the term “scaled” in the system names) to varying degrees with 0 or 1, effectively allowing us to explore multiple points along the precision-recall curve;

- **“Forced”** thresholds, where every identified location is returned with a single SF type label chosen as the SF type with highest model score of the eleven possible types.

B. Location Combination

Given EDL file from our U Penn, we first chose mentions with type “LOC” and “GPE” as location candidates, and then using following two strategies to combine the location items with our text classifier outputs:

- Strategy **“nearest”**: If one segment has sf type prediction, find its location mentions. If current segment doesn’t contain any location mentions, then search the nearest segments, until find location terms or reach beginning and end of the document. Finally, output all {type,location} combinations.
- Strategy **“all”**: Find all the location mentions and sf type predictions in the current document. Output all {type,location} combinations.

Both strategies are used in CP1 submissions, only Strategy “all” is used for CP2 submissions.

In the final submission, the filed “Confidence” is set by the probability scores from text classification models; the filed “Status”, “Resolution”, and “Urgent” are set by their most frequent classes which were calculated through historical statistics.

IX. Conclusion

In evaluation of this year, we use three models to predict SF types, DAN, CACO, and MTL. Among these models, CACO and MTL are novel models, while DAN model is our best performer of last year. To best test the the performance of our systems, we tried different strategies to select the output, through the NI feedback and various thresholds. In addition, we also include locations extracted from the EDL in the final system submissions.

References

[1] Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. Massively multilingual word embeddings. In *arXiv preprint arXiv:1602.01925*, 2016.

[2] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294, 2016.

[3] Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. On adversarial removal of hypothesis-only bias in natural language inference. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 256–262, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Yonatan Belinkov, Adam Poliak, Stuart M. Shieber, Benjamin Van Durme, and Alexander Rush. Dont take the premise for granted: Mitigating artifacts in natural language inference. In *ACL*, 2019.

[5] Alexis Conneau, Guillaume Lample, Marc’Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[7] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1681–1691. ACL, 2015.

[8] Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[9] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[10] Keisuke Sakaguchi and Benjamin Van Durme. Efficient online scalar annotation with bounded support. In *Proceedings of the 56rd Annual Meeting of the Association for Computational Linguistics*. ACL, 2018.

[11] Benjamin Van Durme, Tom Lippincott, Kevin Duh, Deana Burchfield, Adam Poliak, Cash Costello, Tim Finin, Scott Miller, James Mayfield, Philipp Koehn, Craig Harman, Dawn Lawrie, Chandler May, Max Thomas, Julianne Chaloux, Annabelle Carrell, Tongfei Chen, Alex Comerford, Mark Dredze, Benjamin Glass, Shudong Hao, Patrick Martin, Rashmi Sankepally, Pushpendre Rastogi, Travis Wolfe, Ying-Ying Tran, and Ted Zhang. CADET: Computer Assisted Discovery Extraction and Translation. In *Proceedings of the 8th International Conference on Natural Language Processing (IJCNLP): System Demonstrations*, 2017.

[12] Mozhi Zhang, Yoshinari Fujinuma, and Jordan Boyd-Graber. Exploiting cross-lingual subword similarities in low-resource document classification. In *1st Workshop on deep learning approaches for low resource natural language processing (DeepLo)*. ACL, 2018.