Illinois CCG LoReHLT 2016 Named Entity Recognition and Situation Frame Systems

Chen-Tse Tsai, Yangqiu Song, Stephen Mayhew, Mark Sammons, and **Dan Roth**
University of Illinois, Urbana-Champaign
201 N. Goodwin
Urbana, Illinois, 61801
{ctsai12, yqsong, mayhew2, mssammon, danr}@illinois.edu

1 Introduction

This is a report of the University of Illinois LORELEI submissions for Named Entity Recognition (NER) and Situation Frames (SF).

The evaluation, which took place in the Summer of 2016, was based on the Uyghur language, a Turkic language closely related to Uzbek, but geographically centered in northwest China. It is written in Arabic script, and there is no Google translate for the language. There are 2,566 pages in the Uyghur Wikipedia.

We describe our NER and SF approaches separately, starting with the NER.

2 Named Entity Recognition

In the months prior to the evaluation, we had developed two approaches: one exploited parallel text to produce annotations in the Incident Language (IL) and another used a cross-lingual wikifier to directly transfer a model which is trained on high-resource languages (Tsai et al., 2016). However, once the language pack was released, we found that the parallel text was from domains too different to be useful, and the Wikipedia was too small to be effective as anything other than a weak gazetteer. We had to start from scratch.

In our final submission, we incorporated two separate approaches: a rule-based approach, described in §2.5.1 and a model transfer approach, described in §2.5.2. Before we cover those approaches, however, we will describe some tools we built, and how we used the native informants.

2.1 Transliteration

Although Uyghur is officially written in Arabic script called Uyghur Ereb Yëziqi (UEY), there is an auxiliary alphabet called Uyghur Latin Yëziqi (ULY) that consists of Latin characters. Since none of us read Arabic script, we found it convenient to transliterate the Arabic data into ULY. To do this, we used the Chinese-Uyghur dictionary recommended in the `docs/` subdirectory of the language pack. This dictionary came in two forms: Uyghur UEY script, and Uyghur ULY script, where the ULY version is an exact transliteration of the UEY. We used this to learn a high-accuracy transliteration model (Pasternack and Roth, 2009), and applied it to all further data. This was a crucial step in our pipeline. Using the transliteration model, we also converted the UEY-English dictionary into ULY-English format so we could look up words as we examined documents.

2.2 Native Informants

We used the native informants (NIs) for the maximum allowed time of 5 hours. Our approach was to annotate the text with our most up-to-date system we had, and have the NI correct our annotations. That is, the NI never changed the span of a tag, but would only change or remove labels. Prior experiments had suggested that this was much faster than complete annotation, and would give more information overall. We recorded all changes, and propagated them through the rest of the data after the NI had finished.

We interacted with two different NIs, and found that one (Mira) was faster and more knowledgeable than the other (Nur).

Because we did not have much time for training, and because the NIs were not familiar with the task, we had some concerns about quality of annotations. For example, although we tried to explain the task carefully, they had trouble distinguishing between GPE and LOC. One NI wanted to annotate coreferent pronouns in addition to persons. There were also inconsistencies in certain tag spans, such as whether to include "government" or "army" in an ORG.

We interacted with the NI using a custom web interface that allows one to display and annotate documents. We had them share their screen, watched as they annotated, and answered questions as they arose. Since we do not read Arabic script, we found it difficult to do real-time monitoring, even though we were watching.

2.3 Development Set

Using our best current models, corrections from the NI, and a lot of manual annotation of ULY versions of the `Set0` documents by the authors (none of whom speak Uyghur), we created a small development set, which we used mainly for tuning the transfer model. The dev set had 23 documents, 25K tokens, and 3K entities.

2.4 Resources

We found certain provided resources to be an immense help, while others we didn't use at all. We used the Chinese-Uyghur dictionary to learn transliterations, but we didn't use it for translations, even manually. We used the Uyghur-English dictionary extensively, manually and automatically. We used the Uyghur language textbook to a small extent, but in the end it was not very helpful besides giving some guidance for manual annotations. We found that the parallel text was not very useful (especially the Ubuntu language pack section – when we were experimenting, we left this out entirely due to the noise). We did not use the Set S English text, but we used the Uyghur monolingual text from all checkpoints.

In addition to these internal resources, we used some external resources: Wikipedia, Freebase, Wikidata, and Google Translate.

2.5 The Final Model

Our final model was an ensemble of a rule-based system with a model transferred from Uzbek annotated data.

2.5.1 Rule-based Model

We manually created a set rules to extract named entity mentions. Some rules are learned from NIs' annotations, and some are learned from scrutiny of the monolingual text in Set 1 by two of the authors. Note that transliterating Arabic script into ULY allows English speakers to pick up certain clues easily. This rule-based model is supposed to have high precision but low recall. We first introduce the rules in five categories: Gazetteers, Chinese Names, Designators, Other Keywords, and Twitter Handles, and then describe how these rules and resources are applied on the test data.

Gazetteers: We automatically collect Uyghur names from Wikipedia titles and the names in Wikidata. These names are classified into the four entity types according to their FreeBase types. Some of the NI annotations are also added into the gazetteers. We verify the NI annotations manually and only add names that we are reasonably sure about. Moreover, whenever we inspect the monolingual text of Set 1, we add the names that we can recognize in the ULY script. At the time of checkpoint 3, we have 1,792 GPE names, 303 LOC names, 869 PER names, and 281 ORG names. Note that an entity can have multiple entries in the gazetteers with different spellings and suffixes.

Besides a list for each entity type, we have another "black list" which contains words and phrases that should *not* be named entities. This list is mainly created from the NI corrections. Whenever an NI removed a label from our system's predictions, we validated it as best we could and put the mention into the black list if deemed appropriate.

Chinese Names: According to (Li et al., 2011), Chinese Pinying system can be mapped to ULY almost deterministically (the character mapping table in (Li et al., 2011) is not compete). We collect Chinese person names from Wikipedia and convert them to the ULY spellings using the mapping table. This gives us another list of person names.

Designators: Designators are strong clues in finding entities. We collect designators from Wikipedia title mapping and NI annotations. The complete list of designators that we gathered for each entity type is shown in Table 1.

Although words next to designators are usually named entities, many are just common nouns or adjectives. For example, "ancient city" and "beautiful lake". Therefore, we use a large amount of monolingual text to mine the confident names and common modifiers next to designators. For each designator, we count the number of times a word appears before it in the 40k monolingual documents (set0, set1, and set2). The confident names are the tokens that appear more than once

GPE	shehiri, wilayiti, nahiy, yézisi, rayoni, aptonum rayoni, aptonom oblasti, ölki, jumhur
LOC	deryasi, déngiz, köli, tagh, téghi, boghuz, kochisi, aralliri, arili, okyan, qites, teshme, qebristan, mehelli, mehkim, kocha, derya, sharqiratm, baziri, meschi, karidor, baghch, bayawan
PER	ependi, xanim
ORG	ministirliki, uniwérsi, uniwérstéti, partiyisi, teshkilati, bankisi, guruh, idari, partkom, uyushm, doxturxa, komitét, bashqarm, ishxan, birleshm, shirk, parlamént, fondigha

Table 1: Designators of each entity type.

and have PMI (with the corresponding designator) larger than 0.5. The common modifiers for each designator are the words with PMI less than 0.5. Since organization names are usually longer than one token, we also count and compute the PMIs of bigrams before the organization designators. This gives 5,361 confident GPE names, 6,628 LOC names, 3,221 PER names, and 4,175 ORG names.

Besides adding confident names into our gazetteers, at test time, we also annotate the designators along with the words (bigrams for ORG) before them. These annotations are verified by the lists of common modifiers we harvested from the monolingual text and a list of stop words. An annotation is rejected if any word matches any common modifiers or stop words.

We have additional rules for ORG designators: komandi (team), armyis and qoshu (army), jemiyit (association or society), and jamaet xewpsiz (public security). For the team designator, we search for the closest GPE mention before the designator. If the closest GPE mention is within two-token distance from the designator, we make a new ORG mention which spans from the start of the GPE to the designator, and remove the original GPE mention. The same approach is also applied to the army designator. Since there could be multiple GPE and LOC mentions in the names of associations or societies (e.g., French Uyghur Association), We look for consecutive GPE or LOC mentions within 4 tokens before the designator. For "public security" (jamaet xewpsiz or j x), we include the token after and the GPE or LOC mention right before this phrase.

Other Keywords: Since the word Uyghur refers to the ethnic group, it should usually not be tagged. We only tag Uyghur as GPE when it appears before qoshun, rehberli, siyasi rehberl, diyar, and pasport.

Cardinal directions (north, south, west, and east) are usually used to refer to a part of GPE or LOC, and these common modifiers should be included in mentions. After all rules are applied, we look at the word right before each annotation, and include the word as part of the mention if its one of the cardinal directions. If the label of the annotation is LOC or ORG, we do not change the label. If the label is GPE, we change it to LOC unless the cardinal direction is part of a country name (e.g., South Korea and South Africa), since "America" is GPE but "south America" should be tagged as LOC.

Twitter Handles: In tweets, for each token that starts with @ or #, we tag it if it matches entries in the gazetteers. If a token starts with @ but does not match any names in the gazetteers, we tag it as PER, since @ is usually used for mentioning users on Twitter.

The overall algorithm of applying these rules is:

1. Annotate the input text by gazetteers. The gazetteers include Chinese names and the confident names next to designators. If there is any conflict, we always take the longer mention.

2. Apply other rules for merging GPE and ORG designators.

3. Apply the rules for Uyghur and cardinal directions.

4. Apply rules for Twitter handles.

2.5.2 Transfer Model

As a second, complimentary approach, we used the annotated Uzbek data from a prior LORELEI package as the basis for a transfer model. We transfer the model by translating the annotated source data into the target language, and training a target language model.

We began by noticing that Uzbek and Uyghur are very similar, sharing a sizable amount of vocabulary, and several morphological rules. However, while there is a shared vocabulary, the words are usually spelled slightly differently. For example, the word for “southern” is “janubiy” in Uzbek and “jenubiy” in Uyghur. There are a number of spelling variations like this, and several of them are systematic.

To quantify the language similarity, we gathered Uzbek and Uyghur words from respective monolingual corpora, and compared the overlap of vocabularies. We found that there was about 5% overlap using exact string match. Accordingly, a model trained on Uzbek data and tested on our Uyghur dev set would get a negligible score.

We clearly needed a way to translate from Uzbek to Uyghur. One insight is that since the languages are similar, it should be sufficient to translate on a word level. Further, we can prioritize this translation according to high-weight words in an Uzbek trained model, as words important in Uzbek are likely to be important in Uyghur also.

To gather this priority list, we trained a monolingual NER model on Uzbek, using standard NER features, and extracted all word form features with absolute value weight above some threshold. This list contained 11K unique Uzbek words.

Word Translation

We tried several techniques for gathering translations of this list: manual mapping, edit-distance mapping, Google Translate pivot, and cross-lingual CCA with word vectors.

We compared each technique by running the model on the development set described above. For various reasons (small size, annotation from non-native speakers, annotation guideline divergence), we could not completely trust the scores on the dev set, but rather treated them as a benchmark where higher is better, but absolute value has little meaning. Because of this, and also because we were constantly updating the development set in parallel, we do not report scores on the development set.

Manual mapping: We manually translated about 100 words using higher weights in the priority list as a guide. This gave us an immediate increase in score. Where no translation gave a negligible score, this manual translation gave a low, but non-trivial score.

Edit-distance mapping: We gathered a list of Uyghur vocabulary. For each word in the Uzbek weight vector, we found the Uyghur word with the lowest modified edit-distance. We modified the edit-distance algorithm to allow certain substitutions with zero cost, corresponding to the systematic spelling divergences between languages. Then, we filtered the results according to edit distance, where a low distance suggests a good translation. This was a slow process, but yielded 7311 pairs with edit distance of 2 or less. For example, this discovered such pairs as *pokistan-pakistan* and *telegraph-télégraf*. However, it also found such pairs as *gamburg-tembur*, which is not a correct translation. Another limitation is that this would often find “translations” of named entities which should not be translated. However, when we disabled this (easily done because all words are labeled in the Uzbek training set), it did more harm than good.

Google Translate pivot: Using the given Uyghur-English dictionary, we found all entries where the number of Uyghur words equals the number of English words in the definition, and the number is either 1 or 2. For example, *barones-baroness* or *bash bahar-early spring*. In the case of two-word entries, we matched the first Uyghur word to the first English word and the second to the second, to give, for example, *bash-early* and *bahar-spring*. We used Google Translate to translate these English words into Uzbek, effectively getting a transitive translation between Uyghur and Uzbek. This found about 17K pairs.

There are three downsides to this approach. First, since it allows only word-word translations, it fails to take word sense into account. Second, we are really concerned with word usage, not word translations. Some word translations are correct, but are just not used in our annotated corpus. Third, the list of words available in the Uyghur-English dictionary does not necessarily occur in the list of high-weight words from the weight vector in the Uzbek NER model. We may find translations, but they may be only marginally useful.

Cross-lingual CCA with word vectors: Using all the monolingual text available in Uyghur and Uzbek, we learned word vectors. Then we projected these vectors into a shared semantic space using CCA (Faruqui and Dyer, 2014). We used the list of low edit-distance word pairs as the dictionary for the projection. Once all the vectors were in the same space, we found the closest Uyghur word to each Uzbek word in the prioritized list. This produced several interesting pairs, such as *toshkentga-aürümchige*, that is, Tashkent-Urumqi, which are both cities. There were also several cases of non-entities being mapped to entities, and vice-versa.

Since the different word mapping approaches may produce different translations for the same word, we tried two different methods of prioritizing translations. To be clear, if a word had only one translation, we simply used that. If a word had multiple translations, we kept these candidates in a list, and used different methods to select the best translation. For the first method, we ordered the candidate list by perceived trustworthiness of the source of each candidate, such that more trustworthy word maps were first. For example, a word may have a translation from the edit-distance map, and a separate translation from the Google Translate map, and yet another translation from the word vector map. Experiments showed that the edit-distance map was the most trustworthy by a large margin. Next, we used a trigram language model trained on Uyghur monolingual text reorder the candidate list. This was slow but effective.

In the end, we were able to translate about 70% of the Uzbek words into Uyghur.

Features

We used standard NER features, including word forms and affixes. Since there is no capitalization in Arabic script, and consequently no capitalization in ULY script, we did not use capitalization as a feature. We found that Brown clusters were effective, as were gazetteers, gathered as part of the rule-based approach. We experimented with using word vectors as features (i.e. each index is a separate feature) but found that it was not effective. In fact, we found that our best score came when we omitted word form features and vector features. For the former, we hypothesize that this is due to the fact that not all words are translated, and the remaining 30% or so are misleading. For the latter, we suppose that the standard "index as feature" approach to using word vectors is too difficult for linear classifiers.

2.5.3 Ensemble

By looking at the number of predictions and manually examining some predictions on monolingual text, we expect our rule-based model to be high-precision but low-recall, whereas the transfer model can extract more and much better PER mentions. Since our transfer model is developed between checkpoint 2 and 3, we did not have chance to test its performance. It is unclear which model performs better on the other three types by just looking at a few predictions. Based on these properties, we designed four ways to merge the results of two models. Each method is built on the previous method in order to increase the recall.

1. Take all predictions of the rule-based model. If a PER mention overlaps with a PER mention from the transfer model, use the mention from the transfer model.

2. Based on the above, and adding all PER mentions from the transfer model. If a PER mention overlaps with GPE, LOC, or ORG mentions from the rule-based model, we discard the PER mention.

3. Based on the above, and adding all mentions of the other three types from the transfer model. However, a mention will be added only if it does not overlap with any existing mention.

4. Based on 2, and adding all mentions of the other three types from the transfer model. If there is a conflict, take the longer mention.

2.6 Results

The results of each checkpoint are listed in Table 2. The submissions of the first two checkpoints are basically rule-based. However, the rules in these first two versions were much noisier and fewer in number compared to our final rule-based model. For instance, we tagged all phrases beginning with the word "Uyghur" as GPE at the first checkpoint, and most occurrences of the word "Xinjang" were not tagged at the second checkpoint because of a bug in our system.

At checkpoint 3, the rule-based model achieves 56.0 F1, and more surprisingly, the transfer model gets 55.6 F1. Since the transfer model is developed after checkpoint 2, we have no idea of its

Model	F1	mm	ms	sm	ss
Checkpoint 1	29.4	32.2	32.2	32.2	32.2
Checkpoint 2	42.4	45.6	45.6	45.6	45.6
Checkpoint 3					
Rule-Based	56.0	63.4	63.4	63.4	63.5
Transfer	55.6	62.2	62.3	62.2	62.3
Ensemble 1	57.7	63.9	64.0	64.0	64.0
Ensemble 2*	60.2	67.5	67.6	67.6	67.7
Ensemble 3	**60.4**	**68.1**	**68.1**	**68.1**	**68.2**
Ensemble 4	59.1	67.1	67.2	67.2	67.3

Table 2: NER checkpoint results. * indicates our primary submission, **bold** indicates best score in column. mm: overlap-maxmax, ms: overlap-maxsum, sm: overlap-summax, ss: overlap-sumsum

true performance other than some manual inspection. It seems that these two models are different enough for all ensemble methods to work well. The drop from Ensemble 3 to Ensemble 4 indicates that the rule-based model indeed has higher precision. That is, it is better not to change the predictions from the rule-based model (except for PER mentions).

3 Situation Frames

In this section, we introduce our system for identifying Situation Frames. The flowchart of the system is shown in Fig. 1. For each segment (sentence), we first apply a binary classifier to classify the segment whether it is earthquake related topic. Then we identify the entities related to the segment. If we cannot detect any entity inside the segment, we just use the previous entity in the document. If we can detect multiple entities in the segment, we use the same binary earthquake classifier applied to the context of the entity to rank the entities. Then if a selected entity can be associated with the segment, we apply the multi-class classifier to find the type of the situation frames. After that, we output our results.[1]

3.1 System Components

We first introduce the system components we used in situation frame detection.

[1]For Checkpoint 1 and Checkpoint 2, we first detect the entities, and then use the segment covering each detected entity to output situation frames, so the results are not comparable to Checkpoint 3.

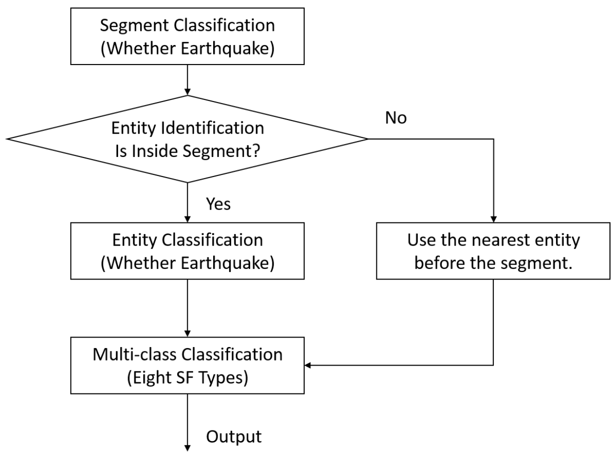
Segment Classification (Whether Earthquake)
Entity Identification Is Inside Segment?
No
Yes
Entity Classification (Whether Earthquake)
Use the nearest entity before the segment.
Multi-class Classification (Eight SF Types)
Output

Figure 1: Situation Frame Identification Framework.

Segment Classification (Whether Earthquake/Disaster). We build a classification component to determine whether a segment is earthquake related. We also build a classifier to classify whether the segment is selected for further processing based on disaster related topics (all the disasters defined in Wikipedia page). The comparison of earthquake based classifier and disaster based classifier is shown in 3.3.

Entity Identification. We use the named entity recognition results described in Section 2. We only use the "GPE" and "LOC" entities for situation frames. There are two cases for each identified segment. First, there is no entity inside the segment. Second, there are some entities inside the segment. For the first case, we simply use the nearest entity before the segment as the identified entity. For the second case, we tried multiple ways on the Chinese parallel corpus, including using the first one appearing in the segment, the last one, the most popular one in the document (entity with most mentions in the document). However, none of the above approaches gave improved results compared to random selection. To improve the entity selection, we build another classifier to rank the entities in order of how related they are to earthquake events. We use the context of the entities (size of window is 10% of the segments for Chinese data, and 5 words behind and ahead for Uyghur data) to represent the entities, then apply the same classifier to the entities, and use the classification score to rank all the entities inside a segment.

Multi-class Classification (Eight SF Types). We

		Chinese				Uyghur	
Model	Parameter	Precision	Recall	F1	LORELEI	LORELEI	Updated
Supervised	3,588 docs	0.258	0.165	0.201	1.311		
Supervised	1,168 docs	0.271	0.165	0.205	1.280	1.420	5.050
Dataless	0.002	0.207	0.234	0.219	1.663	1.380	3.535
Dataless	0.004	0.207	0.176	0.190	1.500	1.152	2.388
Dataless	0.006	0.206	0.141	0.167	1.403	1.054	1.929
Dataless	0.008	0.236	0.134	0.171	1.300	1.087	1.738
Dataless	0.010	0.241	0.126	0.165	1.271	1.062	1.604
Dataless	0.012	0.246	0.112	0.154	1.231	1.065	1.545
Dataless	0.014	0.243	0.101	0.143	1.214	1.069	1.491

Table 3: Situation Frame Identification Results for Checkpoint 3. The parameter for dataless classification is the threshold to cut-off types based on the cosine similarity scores based on English Wikipedia.

follow the annotation guidelines to define eight types of situation frames. The multi-class classifier is applied to each identified segment to label the segment.

3.2 Classifiers

We build two sets of classifiers to classify the segments. One set is based on supervised learning. The other on is based on dataless classification (Chang et al., 2008; Song and Roth, 2014)

3.2.1 Supervised Classifiers

To build the supervised classifier, we use "earthquake" and SF label names as keywords to search Google, Yahoo, and Bing, and collected the top ranked documents for each label. We obtained 3,588 documents for eight labels. However, we found some of the documents are multi-label. Thus, to improve the precision, we remove the documents mentioning label names from multiple SF types, resulting in 1,168 documents. Moreover, we also collected about 10,000 documents from Google news about general topics which are not related to earthquakes. Then we use the 10,000 vs. 1,160 documents to build the binary classifier to classify whether a segment is related to earthquake, and use the 1,160 documents to build the multi-class classifier to identify the types. In this case, we only assign one label per segment based on the multi-class classification results. We used LibLinear to build the classifiers. The features are TFIDF scores of words, where the IDF are computed only based on the corpus we collected.

3.2.2 Dataless Classifiers

The idea of dataless classification is to leverage the label names/descriptions of the categories. Thus, for our situation frame identification problem, we collect the label descriptions differently for the binary and multi-class classifiers. For the binary classifier, we first collected hundreds of documents related to earthquake, and ranked the top 1,000 words from the documents as label descriptions. For multi-class classifier, we simply used the label description shown in the annotation guideline. Then we use the English Wikipedia to build the semantic space for the labels and the segments. For Chinese, since we have the parallel corpus, we use the corresponding English segments for classification. For Uyghur, we use the Uyghur-English dictionary provided in the data to map each word in Uyghur to the English explanation. Then we use cosine similarities computed based on the English Wikipedia used in dataless classification to compare descriptions of types and the segments. Here we use a threshold to cut-off the types that are irrelevant to the segments. In this case, we have a multi-label classifier for each segment of texts.

3.3 Results

For checkpoint 1 and checkpoint 2, we used a different mechanism for situation frame identification, where we first detect the named entities, and then find the corresponding segments to classify them. The best score we got for checkpoint 1 is 1.070 (2.092 for updated score). The best score we got for checkpoint 2 is 0.982 (1.285 for updated score). However, by looking at the Chinese data, we found this mechanism misses a lot of situation frames. Moreover, at checkpoint 2, we compared the earthquake classifier and disaster classifier using dataless classification. With threshold 0.004, the earthquake classifier score is 1.091 while the disaster classifier score is 1.326.

We show our situation frame identification results in checkpoint 3 in Table 3. In the table, we include the results we evaluated on the Chinese parallel corpus as well as the Uyghur data with official scores. The official scores are worse than checkpoints 1 and 2, since with the new classification mechanism, we include more situation frames which may introduce more noise. The supervised classifier using reduced training documents is a little better than when trained with the original 3,588 collected documents. Moreover, by comparing Chinese and Uyghur, we found that by increasing the threshold of dataless classifier, it does not always help for Uyghur data using the original LORELEI score. However, the updated evaluation score shows that it is consistent with the Chinese data. Furthermore, we found that the LORELEI scores are related to precision of the results. However, the F1 score is very low. We look forward to seeing more explanation and evaluation of the difference between the two evaluation metrics.

References

M. Chang, L. Ratinov, D. Roth, and V. Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Proc. of the Conference on Artificial Intelligence (AAAI)*, 7.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. Association for Computational Linguistics.

Jiazheng Li, Kai Liu, Mairehaba Aili, Yajuan Lv, Qun Liu, and Tuergen Yibulayin. 2011. Recognition and translation for chinese names in uighur language. *Journal of Chinese Information Processing*, 25(4):82–88.

J. Pasternack and D. Roth. 2009. Learning better transliterations. In *Proc. of the ACM Conference on Information and Knowledge Management (CIKM)*, 11.

Y. Song and D. Roth. 2014. On dataless hierarchical text classification. In *Proc. of the Conference on Artificial Intelligence (AAAI)*, 7.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016. Cross-lingual named entity recognition via wikification. In *CoNLL*.