Unsupervised Constraint Driven Learning For Transliteration Discovery

Ming-Wei Chang Dan Goldwasser Dan Roth Yuancheng Tu
University of Illinois at Urbana Champaign
Urbana, IL 61801
{mchang21,goldwas1,danr,ytu}@uiuc.edu

Abstract

This paper introduces a novel unsupervised *constraint-driven* learning algorithm for identifying named-entity (NE) transliterations in bilingual corpora. The proposed method does not require any annotated data or aligned corpora. Instead, it is bootstrapped using a simple resource – a romanization table. We show that this resource, when used in conjunction with constraints, can efficiently identify transliteration pairs. We evaluate the proposed method on transliterating English NEs to three different languages - Chinese, Russian and Hebrew. Our experiments show that constraint driven learning can significantly outperform existing unsupervised models and achieve competitive results to existing supervised models.

1 Introduction

Named entity (NE) transliteration is the process of transcribing a NE from a source language to some target language while preserving its pronunciation in the original language. Automatic NE transliteration is an important component in many cross-language applications, such as Cross-Lingual Information Retrieval (CLIR) and Machine Translation(MT) (Hermjakob et al., 2008; Klementiev and Roth, 2006a; Meng et al., 2001; Knight and Graehl, 1998).

It might initially seem that transliteration is an easy task, requiring only finding a phonetic mapping between character sets. However simply matching every source language character to its target language counterpart is not likely to work well as in practice this mapping depends on the context the characters appear in and on transliteration conventions which may change across domains. As a result, current approaches employ machine learning methods which, given enough labeled training data learn how to determine whether a pair of words constitute a transliteration pair. These methods typically require training data and language-specific expertise which may not exist for many languages. In this paper we try to overcome these difficulties and show that when the problem is modeled correctly, a simple character level mapping is a sufficient resource.

In our experiments, English was used as the source language, allowing us to use romanization tables, a resource commonly-available for many languages[1]. These tables contain an incomplete mapping between character sets, mapping every character to its most common counterpart.

Our transliteration model takes a discriminative approach. Given a word pair, the model determines if one word is a transliteration of the other. The features used by this model are character n-gram matches across the two strings. For example, Figure 1 describes the decomposition of a word pair into unigram features as a bipartite graph in which each edge represents an active feature.

We enhance the initial model with constraints, by framing the feature extraction process as a *structured prediction problem* - given a word pair, the set of possible active features is defined as a set of latent binary variables. The contextual dependency be-

[1]The romanization tables available at the Library of Congress website (http://www.loc.gov/catdir/cpso/roman.html) cover more than 150 languages written in various non-Roman scripts

Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL, pages 299–307,
Boulder, Colorado, June 2009.

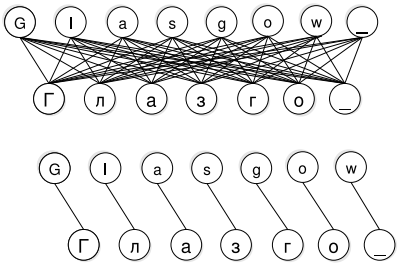

Figure 1: Top: The space of all possible features that can be generated given the word pair. Bottom: A pruned features representation generated by the inference process.

tween features is encoded as a set of constraints over these variables. Features are extracted by finding an assignment that maximizes the similarity score between the two strings and conforms to the constraints. The model is bootstrapped using a romanization table and uses a discriminatively self-trained classifier as a way to improve over several training iterations. Furthermore, when specific knowledge about the source and target languages exists, it can be directly injected into the model as constraints.

We tested our approach on three very different languages - Russian, a Slavic language, Hebrew a Semitic language, and Chinese, a Sino-Tibetan language. In all languages, using this simple resource in conjunction with constraints provided us with a robust transliteration system which significantly outperforms existing unsupervised approaches and achieves comparable performance to supervised methods.

The rest of the paper is organized as follows. Sec. 2 briefly examines more related work. Sec. 3 explains our model and Sec. 4 provide a linguistic intuition for it. Sec. 5 describes our experiments and evaluates our results followed by sec. 6 which concludes our paper.

2 Related Works

Transliteration methods typically fall into two categories: generative approaches (Li et al., 2004; Jung et al., 2000; Knight and Graehl, 1998) that try to produce the target transliteration given a source language NE, and discriminative approaches (Goldwasser and Roth, 2008b; Bergsma and Kondrak, 2007; Sproat et al., 2006; Klementiev and Roth, 2006a), that try to identify the correct transliteration for a word in the source language given several candidates in the target language. Generative methods encounter the Out-Of-Vocabulary (OOV) problem and require substantial amounts of training data and knowledge of the source and target languages. Discriminative approaches, when used to for discovering NE in a bilingual corpora avoid the OOV problem by choosing the transliteration candidates from the corpora. These methods typically make very little assumptions about the source and target languages and require considerably less data to converge. Training the transliteration model is typically done under supervised settings (Bergsma and Kondrak, 2007; Goldwasser and Roth, 2008b), or weakly supervised settings with additional temporal information (Sproat et al., 2006; Klementiev and Roth, 2006a). Our work differs from these works in that it is completely unsupervised and makes no assumptions about the training data.

Incorporating knowledge encoded as constraints into learning problems has attracted a lot of attention in the NLP community recently. This has been shown both in supervised settings (Roth and Yih, 2004; Riedel and Clarke, 2006) and unsupervised settings (Haghighi and Klein, 2006; Chang et al., 2007) in which constraints are used to bootstrap the model. (Chang et al., 2007) describes an unsupervised training of a Constrained Conditional Model (CCM), a general framework for combining statistical models with declarative constraints. We extend this work to include constraints over possible assignments to latent variables which, in turn, define the underlying representation for the learning problem.

In the transliteration community there are several works (Ristad and Yianilos, 1998; Bergsma and Kondrak, 2007; Goldwasser and Roth, 2008b) that show how the feature representation of a word pair can be restricted to facilitate learning a string similarity model. We follow the approach discussed in (Goldwasser and Roth, 2008b), which considers the feature representation as a structured prediction problem and finds the set of optimal assignments (or feature activations), under a set of legitimacy constraints. This approach stresses the importance of interaction between learning and inference, as the model iteratively uses inference to improve the sample representation for the learning problem and uses the learned model to improve the accuracy of the in-

ference process. We adapt this approach to unsupervised settings, where iterating over the data improves the model in both of these dimensions.

3 Unsupervised Constraint Driven Learning

In this section we present our Unsupervised Constraint Driven Learning (**UCDL**) model for discovering transliteration pairs. Our task is in essence a ranking task. Given a NE in the source language and a list of candidate transliterations in the target language, the model is supposed to rank the candidates and output the one with the highest score. The model is bootstrapped using two linguistic resources: a *romanization table* and a set of general and linguistic constraints. We use several iterations of self training to learn the model. The details of the procedure are explained in Algorithm 1.

In our model features are character pairs (c_s, c_t), where $c_s \in C_s$ is a source word character and $c_t \in C_t$ is a target word character. The feature representation of a word pair v_s, v_t is denoted by $F(v_s, v_t)$. Each feature (c_s, c_t) is assigned a weight $W(c_s, c_t) \in R$. In step 1 of the algorithm we initialize the weights vector using the romanization table.

Given a pair (v_s, v_t), a feature extraction process is used to determine the feature based representation of the pair. Once features are extracted to represent a pair, the sum of the weights of the extracted features is the score assigned to the target transliteration candidate. Unlike traditional feature extraction approaches, our feature representation function does not produce a fixed feature representation. In step 2.1, we formalize the feature extraction process as a constrained optimization problem that captures the interdependencies between the features used to represent the sample. That is, obtaining $F(v_s, v_t)$ requires solving an optimization problem. The technical details are described in Sec. 3.1. The constraints we use are described in Sec. 3.2.

In step 2.2 the different candidates for every source NE are ranked according to the similarity score associated with their chosen representation. This ranking is used to "label" examples for a discriminative learning process that learns increasingly better weights, and thus improve the representation of the pair: each source NE paired with its top ranked transliteration is labeled as a positive examples (step 2.3) and the rest of the samples are considered as negative samples. In order to focus the learning process, we removed from the training set all negative examples ruled-out by the constraints (step 2.4). As the learning process progresses, the initial weights are replaced by weights which are discriminatively learned (step 2.5). This process is repeated several times until the model converges, and repeats the same ranking over several iterations.

Input: Romanization table $\mathcal{T} : C_s \rightarrow C_t$, Constraints $\mathcal{C}$, Source NEs: V_s, Target words: V_t

1. Initialize Model

Let $\mathcal{W} : C_s \times C_t \rightarrow R$ be a weight vector.
Initialize $\mathcal{W}$ using $\mathcal{T}$ by the following procedure

$$\forall (c_s, c_t), (c_s, c_t) \in \mathcal{T} \Rightarrow \mathcal{W}(c_s, c_t) = 0,$$
$$\forall (c_s, c_t), \neg((c_s, c_t) \in \mathcal{T}) \Rightarrow \mathcal{W}(c_s, c_t) = -1,$$
$$\forall c_s, \mathcal{W}(c_s, \_) = -1, \forall c_t, \mathcal{W}(\_, c_t) = -1.$$

2. Constraints driven unsupervised training

while *not converged* **do**

1. $\forall v_s \in V_s, v_t \in V_t$, use $\mathcal{C}$ and $\mathcal{W}$ to generate a representation $F(v_s, v_t)$
2. $\forall v_s \in V_s$, find the top ranking transliteration pair (v_s, v_t^*) by solving $v_t^* = \arg\max_{v_t} \text{score}(F(v_s, v_t))$.
3. $D = \{(+, F(v_s, v_t^*)) \mid \forall v_s \in V_s\}$.
4. $\forall v_s \in V_s, v_t \in V_t$, if $v_t \neq v_t^*$ and $\text{score}(F(v_s, v_t)) \neq -\infty$, then $D = D \cup \{(-, F(v_s, v_t))\}$.
5. $\mathcal{W} \leftarrow \text{train}(D)$

end

Algorithm 1: UCDL Transliteration Framework.

In the rest of this section we explain this process in detail. We define the feature extraction inference process in Sec. 3.1, the constraints used in Sec. 3.2 and the inference algorithm in Sec. 3.3. The linguistic intuition for our model is described in Sec. 4.

3.1 Finding Feature Representation as Constrained Optimization

We use the formulation of Constrainted Conditional Models (CCMs) (Roth and Yih, 2004; Roth and Yih, 2007; Chang et al., 2008). Previous work on CCM models dependencies between different decisions in structured prediction problems. Transliteration discovery is a binary classification problem, however,

the underlying representation of each sample can be modeled as a CCM, defined as a set of latent variables corresponding to the set of all possible features for a given sample. The dependencies between the features are captured using constraints.

Given a word pair, the set of all possible features consists of all character mappings from the source word to the target word. Since in many cases the size of the words differ we augment each of the words with a blank character (denoted as '_'). We model character omission by mapping the character to the blank character. This process is formally defined as an operator mapping a transliteration candidate pair to a set of binary variables, denoted as $All\text{-}Features\ (AF)$.

$$AF = \{(c_s, c_t) | c_s \in v_s \cup \{\_\}, c_t \in v_t \cup \{\_\}\}$$

This representation is depicted at the top of Figure 1.

The initial sample representation (AF) generates features by coupling substrings from the two terms without considering the dependencies between the possible combinations. This representation is clearly noisy and in order to improve it we select a subset $F \subset AF$ of the possible features. The selection process is formulated as a linear optimization problem over binary variables encoding feature activations in AF. Variables assigned 1 are selected to be in F, and those assigned 0 are not. The objective function maximized is a linear function over the variables in AF, each with its weight as a coefficient, as in the left part of Equation 1 below. We seek to maximize this linear sum subject to a set of constraints. These represent the dependencies between selections and prior knowledge about possible legitimate character mappings and correspond to the right side of Equation 1. In our settings only *hard constraints* are used and therefore the penalty (ρ) for violating any of the constraints is set to ∞. The specific constraints used are discussed in Sec. 3.2. The **score** of the mapping $F(v_s, v_t)$ can be written as follows:

$$\frac{1}{|v_t|}(\mathcal{W} \cdot F(v_s, v_t) - \sum_{c_i \in \mathcal{C}} \rho c_i(F(v_s, v_t)) \qquad (1)$$

We normalize this score by dividing it by the size of the target word, since the size of candidates varies, normalization improved the ranking of candidates.

The result of the optimization process is a set F of active features, defined in Equation 2. The result of this process is described at the bottom of Figure 1.

$$F^*(v_s, v_t) = \arg\max_{F \subset AF(v_s, v_t)} \text{score}(F). \quad (2)$$

The ranking process done by our model can now be naturally defined - given a source word v_s, and a set of candidates target words $v_t^0, \ldots, v_t^n$, find the candidate whose optimal representation maximizes Equation 1. This process is defined in Equation 3.

$$v_t^* = \arg\max_{v_t^i} \text{score}(F(v_s, v_t^i)). \qquad (3)$$

3.2 Incorporating Mapping Constraints

We consider two types of constraints: language specific and general constraints that apply to all languages. Language specific constraints typically impose a local restriction such as individually forcing some of the possible character mapping decisions. The linguistic intuition behind these constraints is discussed in Section 4. General constraints encode global restrictions, capturing the dependencies between different mapping decisions.

General constraints: To facilitate readability we denote the feature mapping the i-th source word character to the j-th target word character as a Boolean variable a_{ij} that is 1 if that feature is active and 0 otherwise.

- *Coverage* - Every character must be mapped only to a single character or to the blank character. We can formulate this as: $\sum_j a_{ij} = 1$ and $\sum_i a_{ij} = 1$.
- *No Crossing* - Every character mapping, except mapping to blank character, should preserve the order of appearance in the source and target words, or formally - $\forall i, j$ s.t. $a_{ij} = 1 \Rightarrow \forall l < i, \forall k > j, a_{lk} = 0$. Another constraint is $\forall i, j$ s.t. $a_{ij} = 1 \Rightarrow \forall l > i, \forall k < j, a_{lk} = 0$.

Language specific constraints

- *Restricted Mapping*: These constraints restrict the possible local mappings between source and target language characters. We maintain a set of possible mappings $\{c_s \rightarrow \Theta_{c_s}\}$, where $\Theta_{c_s} \subseteq C_t$ and $\{c_t \rightarrow \Theta_{c_t}\}$, where $\Theta_{c_t} \subseteq C_s$. Any feature (c_s, c_t) such that $c_s \notin \Theta_{c_t}$ or $c_t \notin \Theta_{c_s}$ is penalized in our model.

- *Length restriction*: An additional constraint restricts the size difference between the two words. We formulate this as follows: $\forall v_s \in V_s, \forall v_t \in V_t$, if $\gamma|v_t| > |v_s|$ and $\gamma|v_s| > |v_t|$, score$(F(v_s, v_t)) = -\infty$. Although γ can take different values for different languages, we simply set γ to 2 in this paper.

In addition to biasing the model to choose the right candidate, the constraints also provide a computational advantage: a given a word pair is eliminated from consideration when the length restriction is not satisfied or there is no way to satisfy the restricted mapping constraints.

3.3 Inference

The optimization problem defined in Equation 2 is an integer linear program (ILP). However, given the structure of the problem it is possible to develop an efficient dynamic programming algorithm for it, based on the algorithm for finding the minimal edit distance of two strings. The complexity of finding the optimal set of features is only quadratic in the size of the input pair, a clear improvement over the ILP exponential time algorithm. The algorithm minimizes the weighted edit distance between the strings, and produces a character alignment that satisfies the general constraints (Sec. 3.2). Our modifications are only concerned with incorporating the language-specific constraints into the algorithm. This can be done simply by assigning a negative infinity score to any alignment decision not satisfying these constraints.

4 Bootstrapping with Linguistic Information

Our model is bootstrapped using two resources - a romanization table and mapping constraints. Both resources capture the same information - character mapping between languages. The distinction between the two represents the difference in the confidence we have in these resources - the romanization table is a noisy mapping covering the character set and is therefore better suited as a feature. Constraints, represented by pervasive, correct character mapping, indicate the sound mapping tendency between source and target languages. For example, certain n-gram phonemic mappings, such as $r \rightarrow l$ from English to Chinese, are language specific and can be captured by language specific sound change patterns.

Phonemes	Constraints
Vowel	$i \rightarrow y;\ u \rightarrow w; a \rightarrow a$
Nasal	$m \leftrightarrow m;\ \mathbf{m, n} \leftarrow \mathbf{m}$
Approximant	$r \rightarrow l;\ \mathbf{l, r} \leftrightarrow \mathbf{l}$ $l \leftarrow l;\ w \rightarrow h, w, f$ $h, o, u, v \leftarrow w;\ y \rightarrow y$
Fricative	$v \rightarrow w, b, f$ $s \rightarrow s, x, z;\ s, c \leftarrow s$
Plosive	$p \rightarrow b, p;\ p \leftarrow p$ $b \rightarrow b;\ t \leftarrow t$ $t, d \leftarrow d;\ q \rightarrow k$

Table 1: All language specific constraints used in our English to Chinese transliteration (see Sec. 3.2 for more details). Constraints in boldface apply to all positions, the rest apply only to characters appearing in initial position.

These patterns have been used by other systems as features or *pseudofeatures* (Yoon et al., 2007). However, in our system these language specific rule-of-thumbs are systematically used as constraints to exclude impossible alignments and therefore generate better features for learning. We listed in Table 1 all 20 language specific constraints we used for Chinese. There is a total of 24 constraints for Hebrew and 17 for Russian.

The constraints in Table 1 indicate a systematic sound mapping between English and Chinese unigram character mappings. Arranged by manners of articulation each row of the table indicates the sound change tendency among vowels, nasals, approximants (retroflex and glides), fricatives and plosives. For example, voiceless plosive sounds such as *p, t* in English, tend to map to both voiced (such as *b, d*) and voiceless sounds in Chinese. However, if the sound is voiceless in Chinese, its backtrack English sound must be voiceless. This voice-voiceless sound change tendency is captured by our constraints such as $p \rightarrow b, p$ and $p \leftarrow p;\ t \leftarrow t$.

5 Experiments and Analysis

In this section, we demonstrate the effectiveness of constraint driven learning empirically. We start by describing the datasets and experimental settings and then proceed to describe the results. We evaluated our method on three very different target lan-

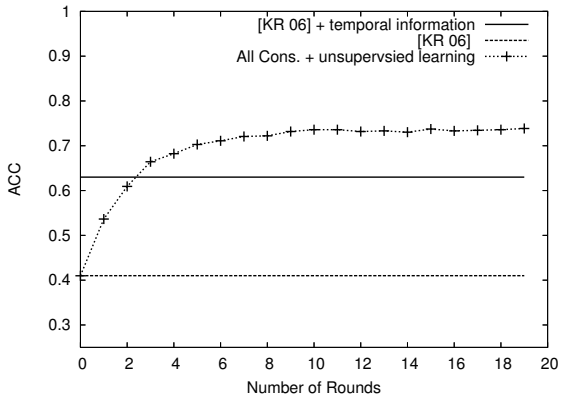

Figure 2: **Comparison between our models and weakly supervised learning methods (Klementiev and Roth, 2006b).** Note that one of the models proposed in (Klementiev and Roth, 2006b) takes advantage of the temporal information. Our best model, the unsupervised learning with all constraints, outperforms both models in (Klementiev and Roth, 2006b), even though we do not use any temporal information.

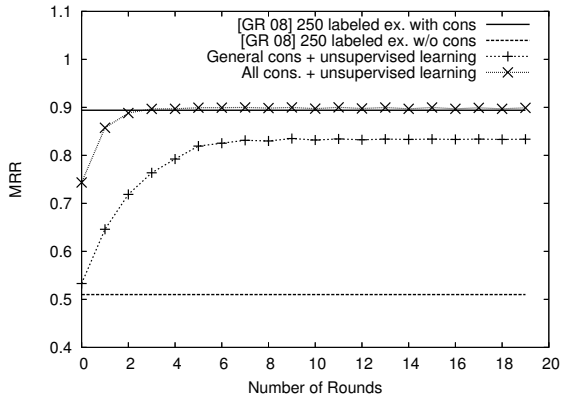

Figure 3: **Comparison between our works and supervised models in (Goldwasser and Roth, 2008b).** We show the learning curves for Hebrew under two different settings: unsupervised learning with general and all constraints. The results of two *supervised models* (Goldwasser and Roth, 2008b) are also included here. Note that our *unsupervised* model with all constraints is competitive to the supervised model with 250 labeled examples. See the text for more comparisons and details.

guages: *Russian, Chinese, and Hebrew*, and compared our results to previously published results.

5.1 Experimental Settings

In our experiments the system is evaluated on its ability to correctly identify the gold transliteration for each source word. We evaluated the system's performance using two measures adopted in many transliteration works. The first one is Mean Reciprocal Rank (MRR), used in (Tao et al., 2006; Sproat et al., 2006), which is the average of the multiplicative inverse of the rank of the correct answer. Formally, Let n be the number of source NEs. Let GoldRank(i) be the rank the algorithm assigns to the correct transliteration. Then, MRR is defined by:

$$\text{MRR} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\text{goldRank}(i)}.$$

Another measure is Accuracy (ACC) used in (Klementiev and Roth, 2006a; Goldwasser and Roth, 2008a), which is the percentage of the top rank candidates being the gold transliteration. In our implementation we used the support vector machine (SVM) learning algorithm with linear kernel as our underlying learning algorithm (mentioned in part 2.5 of Algorithm 1) . We used the package **LIBLINEAR** (Hsieh et al., 2008) in our experiments. Through all of our experiments, we used the 2-norm hinge loss as our loss function and fixed the regularization parameter C to be 0.5.

5.2 Datasets

We experimented using three different target languages *Russian, Chinese, and Hebrew*. We used English as the source language in all these experiments.

The Russian data set[2], originally introduced in (Klementiev and Roth, 2006b), is comprised of temporally aligned news articles. The dataset contains 727 single word English NEs with a corresponding set of 50,648 potential Russian candidate words which include not only name entities, but also other words appearing in the news articles.

The Chinese dataset is taken directly from an English-Chinese transliteration dictionary, derived from LDC Gigaword corpus[3]. The entire dictionary consists of 74,396 pairs of English-Chinese NEs, where Chinese NEs are written in *Pinyin*, a romanized spelling system of Chinese. In (Tao et al., 2006) a dataset which contains about 600 English NEs and 700 Chinese candidates is used. Since the dataset is not publicly available, we created a dataset in a similar way. We randomly selected approximately 600 NE pairs and then added about 100 candidates which do not correspond to any of the English NE

[2]The corpus is available http://L2R.cs.uiuc.edu/∼cogcomp.
[3]http://www.ldc.upenn.edu

Language	UCDL	Prev. works
Rus. (ACC)	73	63 (41) (KR'06)
Heb. (MRR)	0.899	0.894 (GR'08)

Table 2: Comparison to previously published results. UCDL is our method, KR'06 is described in (Klementiev and Roth, 2006b) and GR'08 in (Goldwasser and Roth, 2008b). Note that our results for Hebrew are comparable with a supervised system.

previously selected.

The Hebrew dataset, originally introduced in (Goldwasser and Roth, 2008a), consists of 300 English-Hebrew pairs extracted from Wikipedia.

5.3 Results

We begin by comparing our model to previously published models tested over the same data, in two different languages, Russian and Hebrew. For Russian, we compare to the model presented in (Klementiev and Roth, 2006b), a weakly supervised algorithm that uses both phonetic information and temporal information. The model is bootstrapped using a set of 20 labeled examples. In their setting the candidates are ranked by combining two scores, one obtained using the transliteration model and a second by comparing the relative occurrence frequency of terms over time in both languages. Due to computational tractability reasons we slightly changed Algorithm 1 to use only a small subset of the possible negative examples.

For Hebrew, we compare to the model presented in (Goldwasser and Roth, 2008b), a supervised model trained using 250 labeled examples. This model uses a bigram model to represent the transliteration samples (i.e., features are generated by pairing character unigrams and bigrams). The model also uses constraints to restrict the feature extraction process, which are equivalent to the *coverage* constraint we described in Sec. 3.2.

The results of these experiments are reported using the evaluation measures used in the original papers and are summarized in Table 2. The results show a significant improvement over the Russian data set and comparable performance to the supervised method used for Hebrew.

Figure 2 describes the learning curve of our method over the Russian dataset. We compared our algorithm to two models described in (Klementiev and Roth, 2006b) - one uses only phonetic similarity and the second also considers temporal co-occurrence similarity when ranking the transliteration candidates. Both models converge after 50 iterations. When comparing our model to their models, we found that even though our model ignores the temporal information it achieves better results and converges after fewer iterations. Their results report a significant improvement when using temporal information - improving an ACC score of 41% without temporal information to 63% when using it. Since the temporal information is orthogonal to the transliteration model, our model should similarly benefit from incorporating the temporal information.

Figure 3 compares the learning curve of our method to an existing supervised method over the Hebrew data and shows we get comparable results.

Unfortunately, we could not find a published Chinese dataset. However, our system achieved similar results to other systems, over a different dataset with similar number of training examples. For example, (Sproat et al., 2006) presents a supervised system that achieves a MRR score of 0.89, when evaluated over a dataset consisting of 400 English NE and 627 Chinese words. Our results for a different dataset of similar size are reported in Table 3.

5.4 Analysis

The resources used in our framework consist of - a romanization table, general and language specific transliteration constraints. To reveal the impact of each component we experimented with different combination of the components, resulting in three different testing configurations.

Romanization Table: We initialized the weight vector using a romanization table and did not use any constraints. To generate features we use a modified version of our AF operator (see Sec. 3), which generates features by coupling characters in close positions in the source and target words. This configuration is equivalent to the model used in (Klementiev and Roth, 2006b).

+General Constraints: This configuration uses the romanization table for initializing the weight vector and general transliteration constraints (see Sec. 3.2) for feature extraction.

+All Constraints: This configuration uses language specific constraints in addition to the gen-

Settings		Chinese	Russian	Hebrew
Romanization table		0.019 (0.5)	0.034 (1.0)	0.046 (1.7)
Romanization table	+learning	0.020 (0.3)	0.048 (1.3)	0.028 (0.7)
+Gen Constraints		0.746 (67.1)	0.809 (74.3)	0.533 (45.0)
+Gen Constraints	+learning	0.867 (82.2)	0.906 (86.7)	0.834 (76.0)
+All Constraints		0.801 (73.4)	0.849 (79.3)	0.743 (66.0)
+All Constraints	+learning	**0.889 (84.7)**	**0.931 (90.0)**	**0.899 (85.0)**

Table 3: Results of an ablation study of unsupervised method for three target languages. Results for ACC are *inside* parentheses, and for MRR *outside*. When the learning algorithm is used, the results after 20 rounds of constraint driven learning are reported. Note that using linguistic constraints has a significant impact in the English-Hebrew experiments. Our results show that a small amount of constraints can go a long way, and better constraints lead to better learning performance.

eral transliteration constraints to generate the feature representation. (see Sec. 4).

+Learning: Indicates that after initializing the weight vector, we update the weight using Algorithm 1. In all of the experiments, we report the results after 20 training iterations.

The results are summarized in Table 3. Due to the size of the Russian dataset, we used a subset consisting of 300 English NEs and their matching Russian transliterations for the analysis presented here. After observing the results, we discovered the following regularities for all three languages. Using the romanization table directly without constraints results in very poor performance, even after learning. This can be used as an indication of the difficulty of the transliteration problem and the difficulties earlier works have had when using only romanization tables, however, when used in conjunction with constraints results improve dramatically. For example, in the English-Chinese data set, we improve MRR from 0.02 to 0.746 and for the English-Russian data set we improve 0.03 to 0.8. Interestingly, the results for the English-Hebrew data set are lower than for other languages - we achieve 0.53 MRR in this setting. We attribute the difference to the quality of the mapping in the romanization table for that language. Indeed, the weights learned after 20 training iterations improve the results to 0.83. This improvement is consistent across all languages, after learning we are able to achieve a MRR score of 0.87 for the English-Chinese data set and 0.91 for the English-Russian data set. These results show that romanization table contains enough information to bootstrap the model when used in conjunction with constraints. We are able to achieve results comparable to supervised methods that use a similar set of constraints and labeled examples.

Bootstrapping the weight vector using language specific constraints can further improve the results. They provide several advantages: a better starting point, an improved learning rate and a better final model. This is clear in all three languages, for example results for the Russian and Chinese bootstrapped models improve by 5%, and by over 20% for Hebrew. After training the difference is smaller- only 3% for the first two and 6% for Hebrew. Figure 3 describes the learning curve for models with and without language specific constraints for the English-Hebrew data set, it can be observed that using these constraints the model converges faster and achieves better results.

6 Conclusion

In this paper we develop a constraints driven approach to named entity transliteration. In doing it we show that romanization tables are a very useful resource for transliteration discovery if proper constraints are included. Our framework does not need labeled data and does not assume that bilingual corpus are temporally aligned. Even without using any labeled data, our model is competitive to existing supervised models and outperforms existing weakly supervised models.

7 Acknowledgments

We wish to thank the reviewers for their insightful comments. This work is partly supported by NSF grant SoD-HCER-0613885 and DARPA funding under the Bootstrap Learning Program.

References

S. Bergsma and G. Kondrak. 2007. Alignment-based discriminative string similarity. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 656–663, Prague, Czech Republic, June. Association for Computational Linguistics.

M. Chang, L. Ratinov, and D. Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 280–287, Prague, Czech Republic, Jun. Association for Computational Linguistics.

M. Chang, L. Ratinov, N. Rizzolo, and D. Roth. 2008. Learning and inference with constraints. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, July.

D. Goldwasser and D. Roth. 2008a. Active sample selection for named entity transliteration. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, June.

D. Goldwasser and D. Roth. 2008b. Transliteration as constrained optimization. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 353–362, Oct.

A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*.

U. Hermjakob, K. Knight, and H. Daumé III. 2008. Name translation in statistical machine translation - learning when to transliterate. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 389–397, Columbus, Ohio, June.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. 2008. A dual coordinate descent method for large-scale linear svm. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, pages 408–415, New York, NY, USA. ACM.

S. Jung, S. Hong, and E. Paek. 2000. An english to korean transliteration model of extended markov window. In *Proc. the International Conference on Computational Linguistics (COLING)*, pages 383–389.

A. Klementiev and D. Roth. 2006a. Named entity transliteration and discovery from multilingual comparable corpora. In *Proc. of the Annual Meeting of the North American Association of Computational Linguistics (NAACL)*, pages 82–88, June.

A. Klementiev and D. Roth. 2006b. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages USS,TL,ADAPT, July.

K. Knight and J. Graehl. 1998. Machine transliteration. *Computational Linguistics*, pages 599–612.

H. Li, M. Zhang, and J. Su. 2004. A joint source-channel model for machine transliteration. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 159–166, Barcelona, Spain, July.

H. Meng, W. Lo, B. Chen, and K. Tang. 2001. Generating phonetic cognates to handle named entities in english-chinese cross-langauge spoken document retreival. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop*, pages 389–397.

S. Riedel and J. Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 129–137, Sydney, Australia.

E. S. Ristad and P. N. Yianilos. 1998. Learning string edit distance. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 20(5):522–532, May.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. pages 1–8. Association for Computational Linguistics.

D. Roth and W. Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

R. Sproat, T. Tao, and C. Zhai. 2006. Named entity transliteration with comparable corpora. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 73–80, Sydney, Australia, July.

T. Tao, S. Yoon, A. Fister, R. Sproat, and C. Zhai. 2006. Unsupervised named entitly transliteration using temporal and phonetic correlation. In *Proc. of the Conference on Empirical Methods for Natural Language Processing (EMNLP)*, pages 250–257.

S. Yoon, K. Kim, and R. Sproat. 2007. Multilingual transliteration using feature based phonetic method. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 112–119, Prague, Czech Republic, June.