Constraint-Driven Transliteration Discovery

DAN GOLDWASSER, MING-WEI CHANG, YUANCHENG TU & DAN ROTH

University of Illinois at Urbana-Champaign

Abstract

This paper introduces a novel constraint-driven learning framework for identifying named-entity (NE) transliterations. Traditional approaches to the problem of discovering transliterations depend heavily on correctly segmenting the target and the transliteration candidate and on and aligning these segments.

In this work we propose to formulate the process of aligning segments as a constrained optimization problem. We consider the aligned segments as a latent feature representation and show how to infer an optimal latent representation and how to use it in order to learn an improved discriminative transliteration classifier. Our algorithm is an EM-like iterative algorithm that alternates between an optimization step for the latent representation and a learning step for the classifier's parameters.

We apply this method both in supervised and unsupervised settings, and show that our model can significantly outperform previous methods trained using considerably more resources.[1]

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 & Roth 2006a, Meng et al. 2001, Knight & 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.

[1] This paper extends and unifies our previous work (Goldwasser & Roth 2008b) and (Chang et al. 2009).

Recently, several methods focus on *NE transliteration discovery*, a framework for discovering occurrences of NE in a bilingual corpora. In these settings a classifier is trained to determine if a given pair of words constitute a transliteration pair. The success of these methods depends heavily on correctly segmenting the input words and matching the segments across the two words. Recent discriminative transliteration methods avoid this difficult step and encode the possible alignments as features, and let a discriminative training algorithm assign weights appropriately. Although the relevancy of pairwise features is context sensitive and there are contextual constraints among them, the underlying assumption behind these methods is that a discriminative approach will be sufficient to account for those by weighing features appropriately using sufficient training data. This has been shown to be difficult for language pairs which are very different, such as English and Hebrew (Goldwasser & Roth 2008a).

In this work we combine an explicit alignment process in a discriminative training framework, and directly consider the dependency between correctly aligning the candidate words characters and correct transliteration classification decisions. Our model learns how to correctly align the two words and uses that alignment to learn a better classification model by using the aligned substrings as the feature representation of the word pair. We formulate the alignment process as a constrained optimization process that, given the model parameters (i.e., the local mapping weights), finds the best global alignment between the two words. The flexibility of the model allows us to incorporate prior knowledge about the two languages directly as constraints. After features are extracted, we use a discriminative learning algorithm to update the model, and use the new weight vector to determine the objective function for the optimization based feature extraction.

We apply this method in both supervised and unsupervised settings and consider several different alignment models. We bootstrap the unsupervised model with local information only, corresponding only to a partial mapping between the two character sets, and learn from unlabeled data the complete mapping and the relevant context needed to disambiguate the different possible alignment (or *feature activation*) decisions.

We tested our approach on three very different languages – Russian, a Slavic language, Hebrew, a Semitic language, and Chinese, a Sino-Tibetan language. We show that using our approach we can train a robust transliteration model and outperform existing discriminative method using less resources. Interestingly, when working in an unsupervised setting, we show that using a simple resource – a Romanization table, is enough to bootstrap the model, and outperform supervised methods.

The rest of the paper is organized as follows. Section 2 briefly examines related work. Section 3 explains our model and Section 3.3 provides a

linguistic intuition for it. Section 4 describes our experiments and evaluates our results, and Section 5 concludes.

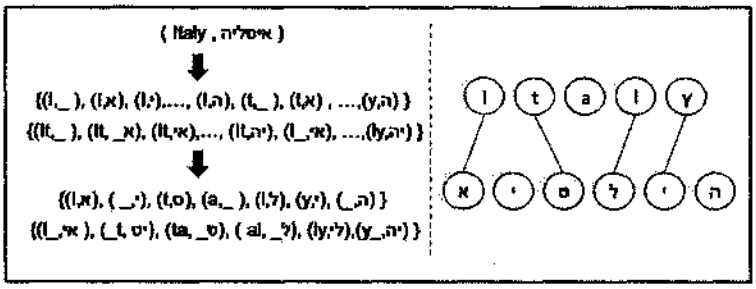

Fig. 1: *Left: The space of all possible features that can be generated given a word pair, and the pruned features representation generated by the inference process. Right: Bipartite graph representation of character unigram alignment corresponding to the generated features*

2 Related work

Transliteration methods typically fall into two categories: generative approaches (Li et al. 2004, Jung et al. 2000, Knight & Graehl 1998) that produce the target transliteration given a source language NE, and discriminative approaches (Goldwasser & Roth 2008b, Bergsma & Kondrak 2007, Sproat et al. 2006, Klementiev & Roth 2006a), that identify the correct transliteration of a word in the source language given several candidates in the target language. Discriminative approaches, when used for discovering NE in a bilingual corpora avoid the Out-Of-Vocabulary 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 & Kondrak 2007, Goldwasser & Roth 2008), or weakly supervised settings with additional temporal information (Sproat et al. 2006, Klementiev & Roth 2006a).

Incorporating knowledge encoded as constraints into learning problems has attracted a lot of attention in the NLP community recently, both in supervised settings (Roth & Yi 2004, Riedel & Clark 2006) and unsupervised settings (Haghighi & Klein 2006, Chang et al. 2007) where constraints are used to bootstrap the model. Chang et al. (2007) describe an unsupervised training of a Constrained Conditional Model (CCM), a general framework for combining statistical models with declarative constraints. We extend their work to include constraints over possible assignments to latent vari-

ables which, in turn, define the underlying representation for the learning problem.

In the transliteration community there are several works (Bergsma & Kondrak 2007, Goldwasser & Roth 2008b, Chang et al. 2009) 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 & 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 inference process. We adapt this approach to unsupervised settings using self-training, where iterating over the data provides a better classification function to label the data for the next training iteration.

3 Constraint-driven transliteration model

In this section we present our Constraint-Driven Transliteration framework. We run an EM-like iterative procedure that alternates between an inference step and a learning step. Inference serves to align the word pair and extract features accordingly; this feature representation is used by the learning algorithm which, in turn, learns the new model parameters thus providing the inference procedure with a better objective function. This process is described in Algorithm 1.

The model presented in this section can be applied in both supervised settings where annotated examples in the form of correct transliteration pairs are available and in unsupervised settings, where this supervision is self generated by the algorithm. In the latter case the initial objective function for the inference process is seeded with a Romanization table – a partial mapping between the source and target character sets. In the rest of this section we describe our framework in detail and explain the differences between the supervised and unsupervised instantiations of the framework.

Transliteration model. Our model works in a *Discovery setting*, where given a source language NE, the model finds its target language counterpart in a document. We use a linear transliteration model, mapping a source language NE and a target language candidate word into a real number – the candidate pair transliteration score. Given a source word NE, v_s, and a list of target words $v_t{}^0 \ldots v_t{}^k$, each candidate target word is paired with the source word NE. These pairs are ranked according to their transliteration score and the model outputs the pair with the highest score.

Features in our model are character n-gram pairs (s_s, s_t), where s_s is a source word character n-gram and s_t is a target word character n-gram. In our experiments we used unigram and bigram pairs. The feature representation of a word pair v_s, v_t is denoted by $F(v_s, v_t)$.

Each feature (s_s, s_t) is assigned a weight $W(s_s, s_t) \in R$, used for deciding the score assigned to that representation. The weight vector is learned using a linear learning algorithm.

In the rest of this section we describe how to obtain $F(\cdot)$ and how to initialize and train $W(\cdot)$.

Initialization. The weight vector W is initialized differently when working in supervised or unsupervised settings. When training data is available, it is used directly to initialize the model's parameters:

$$W(s_s, s_t) = \frac{\#(s_s, s_t)}{\#(s_s)} \times \frac{\#(s_s, s_t)}{\#(s_t)},$$

where $\#(s_s, s_t)$ is the number of occurrences of that feature in the positive sample set. We use a simple feature extraction technique at this initial stage – features are extracted by considering all possible alignments between the source and target word characters and character bigrams. $\#(s_L), L = s, t$ is the number of occurrences of an individual substring, s_L, in any of the features extracted from positive samples in the training set.

In the unsupervised case the model is bootstrapped using a romanization table T. This table contains a partial mapping between the source and target character sets, typically mapping each character to its predominant counterpart. We use this table directly by assigning a uniform zero weight to character level mappings appearing in the table, and a (-1) penalty otherwise:

$$W(s_s, s_t) = \begin{cases} 0 & : (s_s, s_t) \in T \\ -1 & : (s_s, s_t) \notin T \end{cases}$$

Inference-based feature extraction. Given a word pair (v_s, v_t), a feature extraction process is used to determine the feature representation of the pair. Unlike traditional feature extraction approaches, our feature representation function does not produce a fixed feature representation. The feature extraction process is formalized as a constrained optimization problem that captures the interdependencies between the features used to represent the sample, and encodes these dependencies as constraints restricting the space of possible feature activation combinations. That is, obtaining $F(v_s, v_t)$ requires solving an optimization problem, the technical details are described in Section 3.1. The constraints we use are described in Section 3.2.

Prediction. For training to take place each feature representation should be associated with a label. In the supervised case labels are available, in the unsupervised case the model's predictions are converted into labels. The model ranks the different candidates for every source NE according to the similarity score associated with their chosen representation. Each source NE paired with its top ranked transliteration is labeled as a positive example, we leave the other top k ranking pairs unlabeled, and the rest of the samples are considered as negative samples.

Training. The labeled data can now be used directly to train the model and replace the initial weights with weights which are discriminatively learned. This process is repeated several times until the model converges. Over the different training iterations we expect the model to generate a better representation (and a better classification in the unsupervised case), thus allowing the model to improve over multiple training iterations.

Input: Constraints C, Transliteration data: $D = \{(V_s, V_t)\}$

Initialization: Assign weights to table $W : (S_s, S_t) \rightarrow R$
while not converged:
  Inference: Generate a feature representation D^*
  $D^* \leftarrow \bigcup_{(v_s,v_t)\in D} F(v_s, v_t)$. Use C and W to generate $F(v_s, v_t)$
  Prediction: Associate a label with every instance representation $F(v_s, v_t)$
  Training: Train the new transliteration model
  $W \leftarrow \text{train}(D^*)$

Algorithm 1: *Constraint-driven transliteration framework*

In the rest of this section we explain this process in detail. We define the feature extraction inference process in Section 3.1, the constraints used in Section 3.2, the linguistic intuition for our model is described in Section 3.3 and the inference algorithm in Section 3.4.

3.1 *Finding feature representation as constrained optimization*

Deciding if a target word is a transliteration of a source word is a binary classification problem. However, this classification problem is defined over an unknown (or *hidden*) structure. Successfully recovering this structure has high impact on successful classification. We use the formulation of Constrained Conditional Models (CCMs) (Roth & Yi 2004, Roth & Yi 2007, Chiang et al. 2008) to uncover this structure – feature activation decisions

are defined as a set of latent variables and the dependencies between feature activations are captured using constraints over assignments to these variables.

Initial feature representation. Given a word pair, the set of all possible features consists of all possible character bigram and unigram mappings from the source word to the target word. Character omission is modeled by mapping the character to the blank character (denoted as '_'). This representation is depicted in Figure 1. This process is formally defined as an operator mapping a transliteration candidate pair to a set of binary variables, denoted as *All-Features* (*AF*).

$$AF = \{(s_s, s_t) | s_s \in v_s \cup \{\_\}, s_t \in v_t \cup \{\_\}\}$$

Representation decision. The initial sample representation (*AF*) is obtained by coupling substrings from the two terms without considering the dependencies between the possible combinations. To facilitate learning, this representation should be pruned to consider only feature activations corresponding to *legal alignments* of the two words n-grams. This is done by selecting a subset $F \subset AF$ of the possible features, containing a character unigram and bigram alignment of the two words. Figure 1 provides an example of the features generated given a word pair.

The feature extraction process is formulated as a linear optimization problem over a set of binary variables, encoding feature activations in *AF*. 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. The score of the representation $F(v_s, v_t)$ can be written as follows:

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

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 Section 3.2. The result of the optimization process is a set F of active features, defined in Equation 2. The result of this process is described in Figure 1.

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

Transliteration decision. 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.

$$v_t^* = \underset{v_t^i}{\text{argmax}}\ score(F(v_s, v_t^i)) \tag{3}$$

3.2 *Incorporating mapping constraints*

We consider two types of constraints: general constraints that apply to all languages and language specific constraints. General constraints encode global restrictions, capturing the dependencies between different mapping decisions. Language specific constraints typically impose a local restriction such as forcing some of the possible character mapping decisions. The linguistic intuition behind these constraints is discussed in Section 3.3.

General constraints. To facilitate readability we denote the feature activations as a Boolean variables, where a_{ij} denotes a unigram mapping feature activation – where i denotes the i-th source word character, j the j-th target word character. Similarly, $a_{ij,lm}$ denotes a bigram feature activation, mapping the the i-th and l-th source word characters to the j-th and m-th target word characters respectively.

- *Coverage* – Every character unigram (or bigram) must be mapped only to a single character unigram (or bigram), or to the blank character. For the unigram case this can be formally written as:
$$\textstyle\sum_j a_{ij} \leq 1 \text{ and } \sum_i a_{ij} \leq 1\,.$$
- *No crossing* – Every character mapping, except mapping to blank character, should preserve the order of appearance in the source and target words, or formally for the unigram case,
$$\forall i, j\ (a_{ij} = 1) \Rightarrow (\forall l < i,\ \forall k > j, a_{lk} = 0)$$
and
$$\forall i, j\ (a_{ij} = 1) \Rightarrow (\forall l > i,\ \forall k < j, a_{lk} = 0)\,.$$
- *Unigram and bigram alignment consistency* – every bigram and unigram feature decision with overlapping indices should be consistent with each other:
$$\forall i, j, l, m \text{ s.t. } (l = i + 1 \wedge m = j + 1\ ),\ (a_{ij,lm} \leftrightarrow (a_{ij} \wedge a_{lm}))\,.$$

Language-specific constraints.

- *Restricted mapping*: These constraints restrict the possible local mappings between source and target language characters. We maintain a list 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 *Encoding language-specific knowledge as constraints*

Language specific constraints indicate phonetic 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.

These patterns have been used by other systems as features or *pseudo-features* (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 used 20 language specific constraints for English-Chinese pairings, 24 constraints for English-Hebrew and 17 for English-Russian.

3.4 *Efficient inference*

package to solve The optimization problem defined in Equation 2 is formulated as 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 (Section 3.2). Our modifications are only concerned with incorporating the language-specific constraints into the algorithm and ensuring the consistency between unigram and bigram level

features. The first can be done simply by assigning a negative infinity score to any alignment decision not satisfying these constraints. We modify the algorithm to consider at each stage the decision that minimizes the edit cost of both unigram and bigram edit operations, thus ensuring that the resulting alignment is the optimal one and that unigram level mapping decisions do not conflict with bigram level mapping decisions.

4 Experiments and analysis

We evaluated our method empirically in both supervised and unsupervised settings, observing both the overall performance in the classification task and the resources required for achieving this performance. We compared our method to previously published results and show that our model outperforms other models significantly using only a fraction of the resources needed to train previous models. To obtain a better understanding of the model we also describe an ablation study, evaluating the individual contribution of each of the model's elements.

We start by describing the experimental settings and datasets used. We then proceed to describe and analyze the results.

4.1 *Experimental settings*

In our experiments the system is evaluated on its ability to correctly identify the correct transliteration for each source word. The test data consists of pairs of words obtained by pairing every source word NE with all target words. 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 as:

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

Another measure is accuracy (ACC) used in (Klementiev & Roth 2006a, Goldwasser & Roth 2008a), which is the percentage of the candidates the algorithm ranks at the top, that are indeed the correct transliteration.

4.2 *Datasets*

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

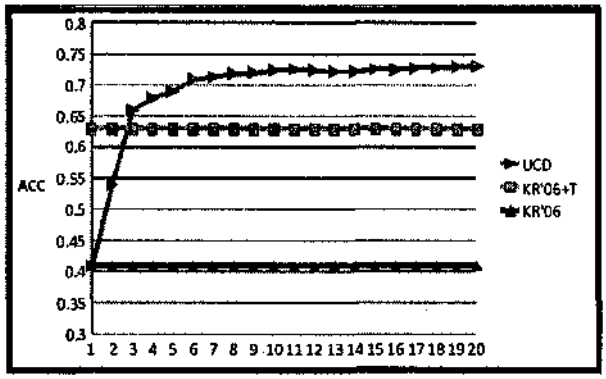

Fig. 2: *Comparison between our model (UCD) and weakly supervised learning methods (Klementiev & Roth 2006b). One of the models proposed in (Klementiev & Roth 2006b) takes advantage of the temporal information. Our best model, the unsupervised learning with all constraints, outperforms both models in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

The Russian data set[2], originally introduced in (Klementiev & 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 100 candidates which do not correspond to any of the English NE previously selected.

The Hebrew dataset, originally introduced in (Goldwasser & Roth 2008a), consists of 550 English-Hebrew transliteration pairs extracted from Wikipedia. In our experiments we used 250 of these NE as training data when working in supervised settings, and the other 300 were used as testing data for both the supervised and unsupervised settings.

[2] The corpus is available at `http://L2R.cs.uiuc.edu/~cogcomp`.

[3] `http://www.ldc.upenn.edu`

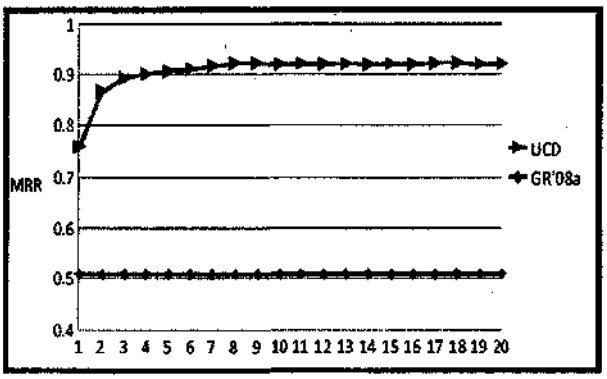

Fig. 3: *Comparison between supervised and unsupervised models, tested on English-Hebrew NE pairs. We show the learning curve for the unsupervised version of our model (UCD), tested on the English-Hebrew dataset. We compare it to the supervised model presented in (Goldwasser & Roth 2008a) (GR'08a). Results show a significant improvement when using our model*

4.3 *Unsupervised settings*

We start by reporting the results obtained by the unsupervised instantiation of our model. We evaluate our model over three different language pairs- Russian, Chinese and Hebrew. Our implementation uses the Support Vector Machine (SVM) learning algorithm with linear kernel as our underlying learning algorithm. We used the package LIBLINEAR (Hsieh et al. 2008) in our experiments.

Our full model uses both unigram and bigram features. However, the supervision signal, obtained from the Romanization table, is limited to unigram features alone. To provide the unsupervised model with a better starting point, the system was trained in two stages – first, using only unigram features, initialized using the Romanization table, and once the model converged we added the bigram features, initialized with a weight of 0. Experiments showed that this training protocol resulted in a considerable improvement.

We begin by comparing our model to previously published models tested over the same data, in two different languages, Russian and Hebrew. The results of these experiments are reported using the evaluation measures used in the original papers and are summarized in Table 1.

To evaluate our performance over the English-Russian dataset, we compare our results to the model presented in (Klementiev & Roth 2006b), a weakly supervised algorithm that uses both phonetic information and

Language	Unsupervised model	Previous work
Russian (ACC)	73%	63% (41%) (KR'06)
Hebrew (MRR)	0.921	0.894 (Supervised model)

Table 1: *Comparison to previously published results. KR'06 is described in (Klementiev & Roth 2006b)*

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, and use only unigram features. The results show a significant improvement for the English-Russian dataset when compared to a previous semi-supervised system, which uses a stronger initial supervision signal. Figure 2 describes the learning curve of our method over the Russian dataset. We compared our algorithm to two models described in (Klementiev & 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 (Klementiev & Roth 2006b), 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.

To evaluate our performance over the English-Hebrew dataset, we compare our performance to the model presented in (Goldwasser & Roth 2008a) a supervised discriminative model trained using 250 labeled examples. This model uses the same feature extraction method as (Klementiev & Roth 2006b), which does not restrict the feature representation of the word pairs. The results show that a significant improvement is obtained when using our model. Figure 3 describes the learning curve of our model over the English-Hebrew dataset.

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) present 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 2.

Settings	Chinese	Russian	Hebrew
Roman. table unig.	0.019 (0.5)	0.034 (1.0)	0.046 (1.7)
Roman. table +learn. unig.	0.020 (0.3)	0.048 (1.3)	0.028 (0.7)
+Gen Const. unig.	0.746 (67.1)	0.809 (74.3)	0.533 (45.0)
+Gen Const. +learn. unig.	0.867 (82.2)	0.906 (86.7)	0.834 (76.0)
+All Const. unig.	0.801 (73.4)	0.849 (79.3)	0.743 (66.0)
+All Const. +learn. unig.	0.889 (84.7)	0.931 (90.0)	0.899 (85.0)
+All Const. +learn. big. (i.)	0.871 (83.4)	0.903 (83.0)	0.884 (83.7)
+All Const. +learn. big. (c.)	**0.902(86.1)**	**0.943(90.4)**	**0.921(87.3)**

Table 2: *Results of an ablation study of the unsupervised method for three target languages. Results for ACC are in parentheses; MRR – outside*

4.4 *Ablation study*

Our system combines several resources and exploits several different intuitions about the transliteration domain. The resources used in our framework consist of a Romanization table and language specific transliteration constraints; in addition our system encodes the dependency between feature activations as general constraints, and it can make use of character unigrams features only, or both character unigrams and bigrams features. To understand the impact of each component we experimented with different combinations of these components, resulting in different testing configurations. The results are presented in Table 2, and explained below. 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.

Romanization Table: We initialized the weight vector using a Romanization table and did not use any constraints. To generate features we used a modified version of our AF operator (see Section 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 & Roth 2006b).

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

+All Constraints: This configuration uses language specific constraints in addition to the general transliteration constraints to generate the feature representation. (see Section 3.3).

+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.

Feature representation: We evaluated our model using unigram and bigram feature models. The Romanization table provides an initial model only for the unigram features, bigram features weights are initially assigned a uniform 0 weight, and learned gradually. We considered three options – using just unigram features, using bigram features in the initial model (denoted *i.* in Table 2) or after the unigram feature model converged (denoted *c.* in Table 2).

Results analysis. The results are summarized in Table 2. 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 in our results for all three languages.

Using the Romanization table directly without constraints results in very poor performance, even after learning. This serves as an indication of the difficulty of the transliteration problem and the difficulty earlier works faced 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 this language pair. 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.

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.

Using bigram features increases the expressivity of the model, as it enables the model to identify the context required to disambiguate character mapping decisions and captures phonetic patterns expressed using several

characters. However using a more expressive model increases the difficulty of the learning problem. When working in an unsupervised setting, a Romanization table may not provide a starting point that is strong enough to bootstrap the extended model. Our experiments indeed show that performance degrades when the extended model is bootstrapped using the Romanization table. However by allowing the model to stabilize using only the unigram features we were able to provide the unsupervised method with a better starting point, resulting in an improved overall performance.

Language	Supervised model	GR'08
Hebrew (MRR)	0.894	0.51

Table 3: *Applying our model in supervised settings, over the English-Hebrew data. Results are compared to Goldwasser & Roth's (2008a) system. Both systems were trained on 250 positive samples*

Fig. 4: *Comparing our method (denoted as SCD in the graph) to (Goldwasser & Roth 2008a) over the English-Hebrew data, using different training sets. Results show that using as little as 10 labeled examples our method can outperform a system trained using 250 labeled examples*

4.5 *Supervised settings*

We also evaluated our system in a supervised setting over the English-Hebrew data. We compare our model to a different discriminative system presented in (Goldwasser & Roth 2008a) evaluated over the same dataset.

Both systems were trained using 250 transliteration pairs, and trained using SNoW (Roth 1998) implementation of the perceptron algorithm.

Our model converged after two iterations over the training data, and was then applied to the testing data, consisting of 300 samples. The results summarized in Table 3 show a significant improvement. Moreover, as can be observed in Figure 4, our model can better use the training data provided – using as little as 10 training examples the resulting model can outperform the baseline model trained using 250 labeled examples. When provided with more data, results improve considerably, while the performance improvement of the baseline model decreases as more training data is added. In Figure 5 we compare the supervised and unsupervised versions

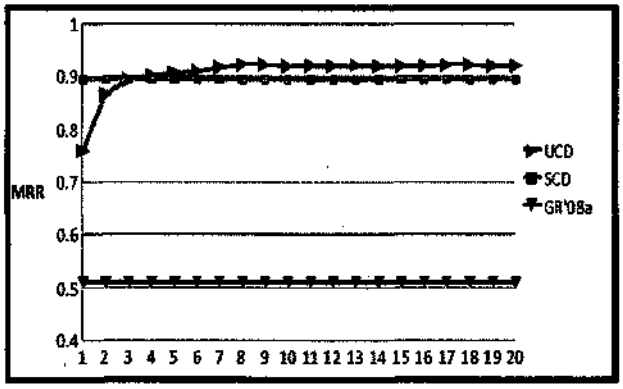

Fig. 5: *Supervised and unsupervised model comparison; tested on English-Hebrew NE pairs. Learning curves for Hebrew under two different settings: unsupervised (UCD) and supervised (denoted SCD). Goldwasser & Roth (2008a) model is also shown (GR'08a). Our unsupervised model outperforms the supervised model, trained on 250 labeled examples*

of our framework over the English-Hebrew dataset. Interestingly, the unsupervised system outperforms the supervised version of the system. This can be explained by the fact that the unsupervised system uses the testing data as training data, allowing it to better adapt to the specific classification instances as it iterates over that data.

5 Conclusion

We introduce a constraint-driven approach for named entity transliteration discovery. This approach identifies the dependency between good representation and successful classification and iterates between the two stages.

We describe how to apply the model in both supervised and unsupervised settings, using only a romanization table. In doing that we show that romanization tables are a very useful resource for transliteration discovery if the proper constraints are enforced. Even without using any labeled data, our model can outperform existing supervised models and weakly supervised models.

Acknowledgements. This work is partly supported by NSF grant SoD-HCER-0613885 and DARPA funding under the Bootstrap Learning Program.

REFERENCES

Bergsma, Shane & Grzegorz Kondrak. 2007. "Alignment-based Discriminative String Similarity". *Annual Meeting of the Association for Computational Linguistics (ACL'2007)*, 656-663. Prague, Czech Republic.

Chang, Ming-Wei, Dan Goldwasser, Dan Roth & Yuancheng Tu. 2009. "Unsupervised, Constraint-driven Learning for Transliteration Discovery". *Annual Meeting of the North American Association for Computational Linguistics (NAACL'2009)*, 299-307. Boulder, Colorado.

Chang, Ming-Wei, Lev Ratinov & Dan Roth. 2007. "Guiding Semi-supervision with Constraint-driven Learning". *Annual Meeting of the Association of Computational Linguistics (ACL'2007)*, 280-287. Prague, Czech Republic.

Chang, Ming-Wei, Lev Ratinov, Nicholas Rizzolo & Dan Roth. 2008. "Learning and Inference with Constraints". *23rd AAAI Conference on Artificial Intelligence (AAAI)*, 1513-1518. Chicago, Illinois. AAAI Press.

Goldwasser, Dan & Dan Roth. 2008a. "Active Sample Selection for Named Entity Transliteration". *Annual Meeting of the Association for Computational Linguistics (ACL'2008)*, 53-56. Columbus, Ohio.

Goldwasser, Dan & Dan Roth. 2008. "Transliteration as Constrained Optimization". *Conference on Empirical Methods for Natural Language Processing (EMNLP-2008)*, 353-362. Waikiki, Honolulu, Hawaii.

Haghighi, Aria & Dan Klein. 2006. "Prototype-driven Learning for Sequence Models". *4th Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2006)*, 320-327. East Stroudsburg, Pennsylvania: Association for Computational Linguistics.

Hermjakob, Ulf, Kevin Knight & Hal Daumé III. 2008. "Name Translation in Statistical Machine Translation – Learning When to Transliterate". *Annual Meeting of the Association of Computational Linguistics (ACL'2008)*, 389-397. Columbus, Ohio. Association for Computational Linguistics.

Hsieh, Cho-Jui, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi & S. Sundararajan. 2008. "A Dual Coordinate Descent Method for Large-scale Linear SVM". *25th International Conference on Machine Learning* (*ICML'08*), 408-415, Helsinki, Finland.

Jung, Sung Young, SungLim Hong & Eunok Paek. 2000. "An English to Korean Transliteration Model of Extended Markov Window". *International Conference on Computational Linguistics* (*COLING-2000*), vol. I, 383-389.

Klementiev, Alexandre & Dan Roth. 2006. "Named Entity Transliteration and Discovery from Multilingual Comparable Corpora". *Annual Meeting of the North American Association for Computational Linguistics* (*NAACL'2006*), 82-88. New York.

Klementiev, Alexandre & Dan Roth. 2006b. "Weakly Supervised Named Entity Transliteration and Discovery from Multilingual Comparable Corpora". *21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics* (*COLING-ACL'2006*), 817-824. Sydney, Australia.

Knight, Kevin & Jonathan Graehl. 1998. "Machine Transliteration". *Computational Linguistics* 24:4.599-612.

Li, Haizhou, Min Zhang & Jian Su. 2004. "A Joint Source-channel Model for Machine Transliteration". *Annual Meeting of the Association for Computational Linguistics* (*ACL'2004*), 159-166. Barcelona, Spain.

Meng, H., W. Lo, B. Chen & K. Tang. 2001. "Generating Phonetic Cognates to Handle Named Entities in English-Chinese Cross-langauge Spoken Document Retreival". *Automatic Speech Recognition and Understanding Workshop*, 389-397.

Riedel, Sebastian & James Clarke. 2006. "Incremental Integer Linear Programming for Non-projective Dependency Parsing". *Conference on Empirical Methods for Natural Language Processing* (*EMNLP*), 129-137. Sydney, Australia.

Roth, Dan & Wen-Tau Yih. 2004. "A Linear Programming Formulation for Global Inference in Natural Language Tasks". *Conference on Natural Language Learning* (*CoNLL-2004*), 1-8. Boston, Massachusetts.

Roth, Dan & Wen-Tau Yih. 2007. "Global Inference for Entity and Relation Identification via a Linear Programming Formulation". *Introduction to Statistical Relational Learning* ed. by Lise Getoor & Ben Taskar, 553-580. MIT Press.

Roth, Dan. 1998. "Learning to Resolve Natural Language Ambiguities: A Unified Approach". *National Conference on Artificial Intelligence* (*AAAI*), 806-813.

Sproat, R., T. Tao & C. Zhai. 2006. "Named Entity Transliteration with Comparable Corpora". *Annual Meeting of the Association for Computational Linguistics* (*ACL'2006*), 73-80. Sydney, Australia.

Tao, Tao, Su-Youn Yoon, Andrew Fister, Richard Sproat & ChengXiang Zhai "Unsupervised Named Entity Transliteration Using Temporal and Phonetic Correlation". *Empirical Methods for Natural Language Processing* (*EMNLP-2006*), 250-257. Sydney, Australia.

Yoon, Su-Youn, Kyoung-Young Kim & Richard Sproat. 2007. "Multilingual Transliteration Using Feature-based Phonetic Method". *Annual Meeting of the Association for Computational Linguistics* (*ACL'2007*), 112-119. Prague, Czech Republic.