Improving G2P from Wiktionary and other (web) resources

Steffen Eger[1]

[1]Text Technology Lab, Goethe University Frankfurt am Main

steeger@em.uni-frankfurt.de

Abstract

We consider the problem of integrating supplemental information strings in the grapheme-to-phoneme (G2P) conversion task. In particular, we investigate whether we can improve the performance of a G2P system by making it aware of corresponding transductions of an external knowledge source, such as transcriptions in other dialects or languages, transcriptions provided by other datasets, or transcriptions obtained from crowd-sourced knowledge bases such as Wiktionary. Our main methodological paradigm is that of *multiple monotone many-to-many alignments* of input strings, supplemental information strings, and desired transcriptions. Subsequently, we apply a discriminative sequential transducer to the multiply aligned data, using subsequences of the supplemental information strings as additional features.

Index Terms: grapheme-to-phoneme conversion, G2P, alignment, Wiktionary, supplemental information, sequential transduction

10.21437/Interspeech.2015-136

1. Introduction

Grapheme-to-phoneme (G2P) conversion is the problem of converting a string of letters into a string of phonetic symbols. It may be an important intermediary step for systems that convert written text to speech. Closely related to G2P are other string transduction problems in natural language processing (NLP) such as transliteration [1], lemmatization [2], and spelling error correction [3]. The classical learning paradigm in each of these settings is to train a model on pairs of strings $\{(\mathbf{x}, \mathbf{y})\}$ and then to evaluate model performance on test data. Thereby, all state-of-the-art modelings we are aware of (e.g., [4, 5, 6, 7, 8]) proceed by first *aligning* the string pairs $(\mathbf{x}, \mathbf{y})$ in the training data. Also, these modelings acknowledge that alignments may typically be of a rather complex nature in which several $\mathbf{x}$ se-

ph	oe	n	i	x
f	i	n	ɪ	ks

Table 1: Sample monotone many-to-many alignment between $\mathbf{x} =$ phoenix and $\mathbf{y} =$ fɪnɪks.

quence characters may be matched up with several $\mathbf{y}$ sequence characters; Table 1 illustrates.

Here,[1] we extend the classical G2P problem by assuming that, at training time, we have available $(M+2)$-tuples of strings $\{(\mathbf{x}, \hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(M)}, \mathbf{y})\}$, where $\mathbf{x}$ is the input string, $\hat{\mathbf{y}}^{(m)}$, for $1 \leq m \leq M$, are *supplemental information* strings, and $\mathbf{y}$ is the desired output string; at test time, we wish to predict $\mathbf{y}$ from $(\mathbf{x}, \hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(M)})$. Generally, we may think of $\hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(M)}$ as arbitrary strings over arbitrary alphabets $\Sigma^{(m)}$, for $1 \leq m \leq M$. For example, $\mathbf{x}$ might be a letter-string and $\hat{\mathbf{y}}^{(m)}$ might be a transliteration of $\mathbf{x}$ in language L_m [10]. Alternatively, as considered here, $\mathbf{x}$ might be a letter input string and $\hat{\mathbf{y}}^{(m)}$ might be the predicted string of phonemes, given $\mathbf{x}$, produced by an (offline) system T_m.

To motivate our approach, consider the situation of training a new G2P system on the basis of, e.g., Combilex [11]. For each letter form in its database, Combilex provides a corresponding phonetic transcription. Now, suppose that, in addition, we can poll an external knowledge source such as Wiktionary for (its) phonetic transcriptions of the respective Combilex letter words as in Table 2. *The central question we want to answer is*: can

Input form	Wiktionary	Combilex
neutrino	njuːtɪiːnoʊ	nutrinF
wooded	wʊdɪd	wUd@d
wrench	ɹɛnt∫	rEn<

Table 2: Input letter words, Wiktionary and Combilex transcriptions.

we train a system using this *additional* information which performs better than the 'baseline' system that ignores the extra information? Clearly, a system with more information should not perform worse than a system with less information (unless, e.g., the additional information is noisy or 'confusing'), but it is a priori not clear at all how the extra information can be included, as [10] note: output predictions may be in distinct alphabets and/or follow different conventions, and simple rule-based conversions may even deteriorate a baseline system's performance.

Our methodological paradigm in this setup is an extension of the discriminative framework for G2P conversion [4, 6, 7]. At training time, we first many-to-many align the training data strings, and then learn a sequence labeling (tagging) model on the aligned data. However, rather than many-to-many aligning *pairs* of strings, we many-to-many align *multiple* strings, and then train a sequential transducer on input strings and the supplemental information strings. Table 3 shows a sample *multiple many-to-many alignment* of input strings, supplemental information strings and desired output string.

This work is structured as follows. Section 2 defines multiple monotone many-to-many alignments. Section 3 surveys related work. Section 4 introduces the data with which we experiment as well as our discriminative baseline method and its extension for our generalized setup. We document our experiments in Section 5 and conclude in Section 6.

2. Multiple many-to-many alignments

We formally define the problem of multiply aligning several strings in a *monotone and many-to-many alignment* manner

[1]The current work complements our related work [9] by providing different and additional experiments.

$\mathbf{x}$ = overcharged	o	v	er	ch	a	r	g	ed
$\hat{\mathbf{y}}^{(\text{PTE})}$ = əʊvətʃaːdʒd	əʊ	v	ə	tʃ	aː	-	dʒ	d
$\hat{\mathbf{y}}^{(\text{CELEX})}$ = @Uv@tSA:dZd	@U	v	@	tS	A:	-	dZ	d
$\mathbf{y}$ = FvBr<ArkD	F	v	Br	<	A	r	k	D

Table 3: Left: Input string $\mathbf{x}$, CELEX and PTE transcriptions, and desired (Combilex) output string $\mathbf{y}$. Right: A multiple many-to-many alignment of $(\mathbf{x}, \hat{\mathbf{y}}^{(\text{PTE})}, \hat{\mathbf{y}}^{(\text{CELEX})}, \mathbf{y})$. Skips are marked by a dash ('-').

(cf. [9]). For notational convenience, in this section, let the N strings to align be denoted by $\mathbf{w}_1, \ldots, \mathbf{w}_N$. Let each $\mathbf{w}_n$, for $1 \leq n \leq N$, be an arbitrary string over some alphabet $\Sigma^{(n)}$. Let $\ell_n = |\mathbf{w}_n|$ denote the length of $\mathbf{w}_n$. Moreover, assume that a set $S \subseteq \prod_{n=1}^{N} \{0, \ldots, \ell_n\} \backslash \{\mathbf{0}_N\}$ of *allowable steps* is specified. The set S defines the types of valid 'many-to-many match-up operations', i.e., when $(s_1, \ldots, s_N) \in S$, then this means we may match up sub-sequences of $\mathbf{w}_1$ of length s_1, sub-sequences of $\mathbf{w}_2$ of length s_2, ..., sub-sequences of $\mathbf{w}_N$ of length s_N.

It is convenient to define a monotone multiple many-to-many alignment of $\mathbf{w}_1, \ldots, \mathbf{w}_N$ as an $N \times k$ (for some $k \geq 1$) nonnegative integer matrix $\mathbf{A} := \mathbf{A}_{\mathbf{w}_1,\ldots,\mathbf{w}_N} \in \mathbb{Z}^{N \times k}$ satisfying $\mathbf{A}\mathbb{1}_k = (\ell_1, \ldots, \ell_N)^{\intercal}$, i.e., the rows of $\mathbf{A}$ sum up to the lengths of the respective strings, and where each column of $\mathbf{A}$ lies in S. For any such alignment, we let $(\mathbf{w}_{i,1}, \ldots, \mathbf{w}_{i,k})$ be the corresponding *induced segmentation* of string $\mathbf{w}_i$, for $i = 1, \ldots, N$.

Example. *For appropriate S (including, e.g., steps $(1,2,2,1), (1,1,1,1)$, etc.), the alignment of $\mathbf{x}$, $\hat{\mathbf{y}}^{PTE}$, $\hat{\mathbf{y}}^{CELEX}$, and $\mathbf{y}$ shown in Table 1 may be represented by the 4×8 matrix $\mathbf{A} = \begin{pmatrix} 1 & 1 & 2 & 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 1 & 2 & 2 & 0 & 2 & 1 \\ 2 & 1 & 1 & 2 & 2 & 0 & 2 & 1 \\ 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$. The corresponding induced segmentations are (o,v,er,ch,a,r,g,ed), (əʊ,v,ə,tʃ,aː,-,dʒ,d), (@U,v,@,tS,A:,-,dZ,d), (F,v,Br,<,A,r,k,D).*

Let $\mathcal{A}_S = \mathcal{A}_S(\mathbf{w}_1, \ldots, \mathbf{w}_N)$ denote the class of all multiple alignments of $\mathbf{w}_1, \ldots, \mathbf{w}_N$. For an alignment $\mathbf{A} \in \mathcal{A}_S$, denote by $f(\mathbf{A})$ the *score* of alignment $\mathbf{A}$ under *alignment model* f, where $f : \mathcal{A}_S \to \mathbb{R}$. We now investigate a solution to the *problem of finding the alignment with maximal score* under a particular choice of alignment model f, i.e., we search to solve

$$\max_{\mathbf{A} \in \mathcal{A}_S(\mathbf{w}_1,\ldots,\mathbf{w}_N)} f(\mathbf{A}). \tag{1}$$

In particular, we assume that $f(\mathbf{A})$ is the score

$$f(\mathbf{A}) = \sum_{j=1}^{k} \mathsf{sim}_1(\mathbf{w}_{1,j}, \ldots, \mathbf{w}_{N,j}) \tag{2}$$

for an appropriate real-valued similarity function sim_1 evaluating similarity of a 'column' of N sub-sequences. We call the model f in (2) a *unigram model* because $f(\mathbf{A})$ is the sum of the similarity scores of the matched-up sub-sequences $(\mathbf{w}_{1,j}, \ldots, \mathbf{w}_{N,j})$, ignoring context. Due to this independence assumption, solving maximization problem (1) under specification (2) is straightforward via a dynamic programming (DP) recursion. To do so, define by $M_{S,\mathsf{sim}_1}(i_1, i_2, \ldots, i_N)$ the score of the best alignment, under alignment model $f = \sum \mathsf{sim}_1$ and set of steps S, of $(\mathbf{w}_1(1:i_1), \ldots, \mathbf{w}_N(1:i_N))$.[2] Then, $M_{S,\mathsf{sim}_1}(i_1, \ldots, i_N)$ is equal to

$$\max_{(j_1,\ldots,j_N) \in S} \{M_{S,\mathsf{sim}_1}(i_1 - j_1, \ldots, i_N - j_N) + \mathsf{sim}_1\big(\mathbf{w}(i_1 - j_1 + 1 : i_1), \ldots, \mathbf{w}(i_N - j_N + 1 : j_N)\big)\}. \tag{3}$$

This recurrence directly leads to a DP algorithm [9] for computing the *score* of the best alignment of $(\mathbf{w}_1, \ldots, \mathbf{w}_N)$; the actual alignment can be found by storing pointers to the maximizing steps taken. If similarity evaluations $\mathsf{sim}_1(\mathbf{w}_{1,j}, \ldots, \mathbf{w}_{N,j})$ are thought of as taking constant time, this algorithm's run time is $\mathcal{O}(\prod_{n=1}^{N} \ell_n \cdot |S|)$. When $\ell = \ell_1 = \cdots = \ell_n$ and $|S| = \ell^N - 1$ ('worst case' size of S), then the algorithm's runtime is thus $\mathcal{O}(\ell^{2N})$.[3]

3. Related work

Monotone alignments have a long tradition, both in NLP and bioinformatics. The classical Needleman-Wunsch algorithm [12] computes the optimal alignment between two sequences when only single character matches, mismatches, and skips are allowed. It is a special case of the unigram model (2) in optimization problem (1) for which $N = 2$, $S = \{(1,0), (0,1), (1,1)\}$ and sim_1 takes on values from $\{0, -1\}$, depending on whether compared input sub-sequences match or not. This alignment specification is equivalent to the edit distance problem [13] in which the minimal number of insertions, deletions and substitutions is sought that transforms one string into another. *Substring-to-substring edit operations* — or equivalently, (monotone) many-to-many alignments — have appeared in the NLP context, e.g., in [14], [3], [4], [5], [7], or, significantly earlier, in [15], [16]. *Learning* edit distance/monotone alignments in an *unsupervised* manner has been the topic of, e.g., [17], [18], besides the works already mentioned. All of these approaches are special cases of our unigram model outlined in Section 2 — i.e., they consider particular S (most prominently, $S = \{(1,0), (0,1), (1,1)\}$) and/or restrict attention to only $N = 2$ strings.[4]

Alignments between multiple sequences, i.e., multiple sequence alignment, has also been an issue both in NLP (e.g., [19], [20]) and bioinformatics (e.g., [21]). However, few works consider multiple *many-to-many* sequence alignments (see our own work, [9], and [22] for exceptions), a seemingly straightforward, yet potentially extremely useful generalization in several NLP applications.

From an application perspective, the two approaches most closely related to us are [10] and [23]. In [10], a G2P system is boosted by considering transliterations of the input string in other languages. The approach is to let the G2P system output its n-best predictions for the input string and then to apply an SVM reranker trained on the supplemental information

[2] We denote by $\mathbf{x}(a:b)$ the substring $x_a x_{a+1} \cdots x_b$ of the string $x_1 x_2 \cdots x_t$.

[3] Note that this quickly becomes untractable as N, the number of strings to align, becomes large.

[4] In [18], context influences alignments, so that the approach goes beyond the unigram model sketched in (2), but there, too, the focus is on the situation $N = 2$ and $S = \{(1,0), (0,1), (1,1)\}$.

strings to rerank these transcriptions. Re-ranking methods are generally suboptimal, as [23] point out, because they do not combine predictions at the level of substructures. Moreover, for reranking to be successful, the correct transcription must be in the n-best list. In [23], general ensemble methods for structured prediction are investigated, but the assumption is that the strings to combine share the *same number* of substructures, an assumption that certainly does not hold for G2P transcriptions of arbitrary distinct systems (as, e.g., exemplified in Tables 2 and 3). In fact, our multiple many-to-many alignment approach precisely segments the multiple strings to combine in such a way that the resulting aligned strings satisfy the 'equal number of substructure' condition, so that each of the approaches suggested in [23] would be applicable in our situation *after the multiple many-to-many alignment* step has been performed.

4. Data and systems

4.1. Data

For our experiments below, we consider the following datasets for English: the General American (GA) variant of Combilex [11], CELEX [24], which provides received pronunciation (RP) transcriptions, Wiktionary (both GA and RP variants), and PhotoTransEdit[5] (PTE) in an RP variant. In addition, we consider German and Dutch transcriptions as available from CELEX. Each of these datasets has own phonetic conventions and its own phonetic inventary. To illustrate this distinctiveness, sizes of phonetic inventaries across the datasets are shown below. In particular, we observe that Wiktionary has many more phonetic symbols than the other datasets, a finding that we attribute to its crowd-sourced nature and lacking of normalization.

	Dataset	$\lvert\Sigma\rvert$
English	Combilex	54
	CELEX	40
	$\text{Wiktionary}_{\text{GA}}$	107
	$\text{Wiktionary}_{\text{RP}}$	116
	PTE	57
German (DE)	CELEX	55
Dutch (NL)	CELEX	43

4.2. Systems

BASELINE Our baseline system is a linear-chain conditional random field model (CRF)[6] [25] which we apply in the manner indicated in the introduction: after many-to-many aligning the training data as in Table 1, at *training time*, we use the CRF as a tagging model that is trained to label each input character sub-sequence with an output character sub-sequence. As *features* for the CRF, we use all n-grams (up to size 6) of subsequences of $\mathbf{x}$ that fit inside a window of size 5 centered around the current sub-sequence (*context features*). We also include *linear-chain features* which allow previously generated output character sub-sequences to influence current output character sub-sequences. In essence, our baseline model is a standard discriminative approach to G2P. It is, by and large, the same approach as described in [7], except that we do not include joint n-gram features. At *test time*, we first segment a new input string $\mathbf{x}$ and then apply the CRF. Thereby, we train the segmentation module on the segmented $\mathbf{x}$ sequences, as available from the aligned training data.

[5] Available from http://www.photransedit.com/.

[6] We made use of the CRF++ package available at https://code.google.com/p/crfpp/.

BASELINE+X As competitors for the baseline system, we introduce systems that rely on the predictions of one or several additional (black box/offline) systems. *At training time*, we first multiply many-to-many align the input string $\mathbf{x}$, the predictions $\hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(M)}$ and the true transcription $\mathbf{y}$ as illustrated in Table 3 (see also below). Then, as for the baseline system, we train a CRF to label each input character sub-sequence with the corresponding output character sub-sequence. However, this time, the CRF has access to the sub-sequence suggestions (as the alignments indicate) produced by the offline systems. As *features* for the so extended models, we additionally include context features for all predicted strings $\hat{\mathbf{y}}^{(m)}$ (all n-grams in a window of size 3 centered around the current sub-sequence prediction). To restrict computational costs, we also reduce window size to 3 for the input string. We also include a joint feature firing on the tuple of the current sub-sequence value of $\mathbf{x}$, $\hat{\mathbf{y}}^{(1)}, \ldots, \hat{\mathbf{y}}^{(M)}$.

PHONETISAURUS Finally, we also indicate the performance of Phonetisaurus (PHON.) [8] on our datasets, as a state-of-the-art reference. Phonetisaurus appears to perform on par or better than most competitor systems in G2P and trains and decodes orders of magnitudes faster [8].

4.3. Alignments

To generate multiple many-to-many alignments for the BASELINE+X systems, we specify sim_1 in Eq. (2), as $\mathsf{sim}_1(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(1)}, \ldots, \hat{\mathbf{y}}_i^{(M)}, \mathbf{y}_i) =$

$$\Big(\sum_{m=1}^{M} \mathsf{psim}(\mathbf{x}_i, \hat{\mathbf{y}}_i^{(m)})\Big) + \mathsf{psim}(\mathbf{x}_i, \mathbf{y}_i),$$

where psim is a pair-similarity function. The advantage with this specification is that the similarity of a tuple of subsequences is defined as the sum of *pairwise* similarity scores, which we can directly estimate from pairwise alignments of $(\mathbf{x}, \hat{\mathbf{y}}^{(m)})$ that an off-the-shelf pairwise aligner can produce (we use the Phonetisaurus aligner for this). We set $\mathsf{psim}(\mathbf{u}, \mathbf{v})$ as log-probability of observing the tuple $(\mathbf{u}, \mathbf{v})$ in the training data of pairwise aligned sequences.

5. Experiments

Throughout, we use as accuracy measures for all our systems **word accuray** (WACC). Word accuracy is defined as the number of correctly transcribed strings among all transcribed strings in a test sample. Moreover, in system combination, we always train and test on input strings for which transcriptions for all systems are available (which often results in few strings to train and test on).

5.1. Wiktionary ↔ Combilex

We start out with transcribing input letter strings with the help of Wiktionary into Combilex ($\longrightarrow$ Combilex) and vice versa ($\longrightarrow$ Wiktionary) for training set sizes of 2,000, 5,000 and 10,000. We evaluate on disjoint test sets of size about 3,500. Table 4 shows that both BASEL.+Wik_{GA} and BASEL.+Combilex perform much better than the baseline method itself in predicting Combilex resp. Wiktionary (GA) transcriptions. Moreover, performance differences decrease with training set size. For instance, at 2,000 training strings, BASEL.+Combilex has about 95% higher WACC than BASELINE (73% higher than PHON.), while at 10,000 training strings, this difference decreases to about 28% (25%). For space reasons, we omit results for the

directions (⟶ Wiktionary$_{GA}$) as well as (⟶ Combilex) when the added system is Wiktionary$_{RP}$, noting that they are very similar to the ones outlined.

	⟶ Combilex		
	PHON.	BASEL.	BASEL.+Wik$_{GA}$
2,000	41.80	38.44	**60.71**
5,000	55.70	51.69	**65.81**
10,000	62.47	58.97	**67.30**
	⟶ Wiktionary$_{RP}$		
	PHON.	BASEL.	BASEL.+Combilex
2,000	28.91	25.64	**50.01**
5,000	37.51	36.30	**51.69**
10,000	41.52	40.56	**51.91**

Table 4: WACC in % for different settings. Test sizes 3,500.

5.2. PTE ↔ Combilex ↔ CELEX

Next, we consider the three datasets PTE, Combilex and CELEX in conjunction. Table 5 shows the results. Again, we find that the extended baseline systems perform much better than the systems without supplemental information strings. For example, at training set size 1,099, WACC for BASEL.+Combilex is 139% higher than the baseline system's for the direction (⟶ PTE). In addition, we observe that adding more systems may be beneficial, but apparently this requires their performance not to be too dissimilar. For example, for the direction (⟶ Combilex), BASEL.+CELEX+PTE has about 3% higher WACC than BASEL.+CELEX when BASEL.+CELEX has a very similar WACC as BASEL.+PTE. This relationship does not hold when the two WACCs are very different (as is the case for 1,099 training data points).

	⟶ CE		⟶ Co		⟶ PTE	
	1,099	2,687	1,099	2,687	1,099	2,687
PHON.	34.17	46.32	38.01	50.35	27.44	43.68
B.	25.11	40.78	31.34	45.75	22.21	40.78
B.+Co	**60.03**	64.00			**53.11**	67.90
B.+CE			**57.68**	63.47	46.78	63.60
B.+PTE	44.74	58.02	50.22	62.80		
B.+Co+CE					51.44	**68.01**
B.+Co+PTE	50.66	**64.25**				
B.+CE+PTE			52.14	**65.23**		

Table 5: WACC in % for different settings. Test set size 1,500.

5.3. DE ↔ NL

Having observed that adding supplemental information strings in another *dialect* (RP vs. GA) may be beneficial, we ask whether we can also boost performance of a G2P system when knowing the input string's transcription in another *language*. We choose German and Dutch because their pronunciation may be considered relatively similar. Since the German and Dutch CELEX data sets virtually have no common input strings, we create the German pronunciations of Dutch words by training a high-performing Phonetisaurus system for German (WACC of close to 95%) on CELEX and have it phonetically transcribe Dutch words, both in the training and test data. As Table 6 shows, the answer to our question is both positive and negative. When training set size is very small, knowing German transcriptions helps the Dutch system, but when training set size is large enough ($>$ 1,000 training strings), then the German transcriptions of Dutch words seem to confuse the learner. In other words, the German transcriptions seem to contribute more noise than useful information, given enough Dutch training data.

	⟶ NL		
	PHON.	BASEL.	BASEL.+DE
500	45.76	38.70	**46.28**
1,000	**60.11**	55.38	54.17
2,000	**72.17**	67.16	58.68
5,000	**82.45**	80.94	68.78

Table 6: WACC in % for different settings. Test size size 5,000.

5.4. Combilex ↔ CELEX

Finally, we investigate whether the last observed effect obtains whenever training data is 'large enough'. In other words, do the supplemental information strings always have the potential to confuse the baseline system in case that the conventions of the target transcription system differs from that of supplemental information strings? Table 7 seems to indeed corroborate this conjecture. We find the same 'transition point' in WACC as in the (DE↔NL) setting, but this time the transition occurs at much larger training set sizes (about 30,000 string pairs).

	⟶ Combilex		
	PHON.	BASEL.	BASEL.+CELEX
2,000	49.30	41.61	**64.10**
5,000	61.41	58.30	**69.75**
10,000	71.89	68.35	**74.76**
30,000	**85.46**	82.25	79.16
50,000	**90.23**	86.68	80.54

Table 7: WACC in % for different settings. Test set size 5,500.

6. Conclusions

We have generalized the task description for string transduction to include supplemental information strings. Moreover, we have suggested multiple many-to-many alignments — and a subsequent standardly extended discriminative approach — for solving G2P in this generalized setup. We have shown that adding supplemental information strings via our approach may substantially increase performance measures. However, we have also found that the advantage of this approach is largest in case of little training data. In other words, our approach may be very helpful in a 'low-resource setting' where a new transcription system, for which little training data is available, may be boosted based on existing rich resources. We have found that when plenty of training data is available, the supplemental information strings may confuse a baseline system, rather than boost it. In future work, we intend to investigate the last effect in more detail, attempting to overcome it. Also, it may be worthwhile to consider alternative G2P model classes in our extended string transduction setting, such as *joint models* rather than *discriminative* ones, as we have scrutinized here.

The multiple many-to-many alignment toolkits discussed here will be made available via the author's homepage.

7. References

[1] T. Sherif and G. Kondrak, "Substring-based transliteration." in *ACL*, J. A. Carroll, A. van den Bosch, and A. Zaenen, Eds. The Association for Computational Linguistics, 2007.

[2] M. Dreyer, J. Smith, and J. Eisner, "Latent-variable modeling of string transductions with finite-state methods." in *EMNLP*. ACL, 2008, pp. 1080–1089.

[3] E. Brill and R. C. Moore, "An improved error model for noisy channel spelling correction," in *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ser. ACL '00. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000, pp. 286–293. [Online]. Available: http://dx.doi.org/10.3115/1075218.1075255

[4] S. Jiampojamarn, G. Kondrak, and T. Sherif, "Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion," in *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. Rochester, New York: Association for Computational Linguistics, April 2007, pp. 372–379. [Online]. Available: http://www.aclweb.org/anthology/N/N07/N07-1047

[5] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion." *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[6] S. Jiampojamarn, C. Cherry, and G. Kondrak, "Joint processing and discriminative training for letter-to-phoneme conversion," in *Proceedings of ACL-08: HLT*. Columbus, Ohio: Association for Computational Linguistics, June 2008, pp. 905–913. [Online]. Available: http://www.aclweb.org/anthology/P/P08/P08-1103

[7] ——, "Integrating joint n-gram features into a discriminative training framework." in *HLT-NAACL*. The Association for Computational Linguistics, 2010, pp. 697–700.

[8] J. R. Novak, N. Minematsu, and K. Hirose, "WFST-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding," in *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing*. Donostia–San Sebastin: Association for Computational Linguistics, July 2012, pp. 45–49. [Online]. Available: http://www.aclweb.org/anthology/W12-6208

[9] S. Eger, "Multiple many-to-many sequence alignment for combining string-valued variables: A g2p experiment," in *ACL*. Association for Computational Linguistics, 2015, accepted.

[10] A. Bhargava and G. Kondrak, "Leveraging supplemental representations for sequential transduction." in *HLT-NAACL*. The Association for Computational Linguistics, 2012, pp. 396–406.

[11] K. Richmond, R. A. J. Clark, and S. Fitt, "Robust LTS rules with the Combilex speech technology lexicon." in *INTERSPEECH*. ISCA, 2009, pp. 1295–1298.

[12] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, Mar. 1970. [Online]. Available: http://dx.doi.org/10.1016/0022-2836(70)90057-4

[13] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

[14] S. Deligne, F. Yvon, and F. Bimbot, "Variable-length sequence matching for phonetic transcription using joint multigrams." in *EUROSPEECH*. ISCA, 1995.

[15] E. Ukkonen, "Algorithms for approximate string matching," *Information and Control*, vol. 64, pp. 100–118, 1985.

[16] J. Véronis, "Computerized correction of phonographic errors." *Computers and the Humanities*, vol. 22, no. 1, pp. 43–56, 1988.

[17] E. S. Ristad and P. N. Yianilos, "Learning string-edit distance." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 5, pp. 522–532, 1998.

[18] R. Cotterell, N. Peng, and J. Eisner, "Stochastic contextual edit distance and probabilistic FSTs," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Baltimore, Jun. 2014, 6 pages.

[19] M. A. Covington, "Alignment of multiple languages for historical comparison," in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*. Montreal, Quebec, Canada: Association for Computational Linguistics, August 1998, pp. 275–279. [Online]. Available: http://www.aclweb.org/anthology/P98-1043

[20] A. Bhargava and G. Kondrak, "Multiple word alignment with Profile Hidden Markov Models," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*. Boulder, Colorado: Association for Computational Linguistics, June 2009, pp. 43–48. [Online]. Available: http://www.aclweb.org/anthology/N/N09/N09-3008

[21] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998. [Online]. Available: http://www.amazon.com/Biological-Sequence-Analysis-Probabilistic-Proteins/dp/0521629713

[22] L. Yao and G. Kondrak, "Joint generation of transliterations from multiple representations," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, May–June 2015, pp. 943–952. [Online]. Available: http://www.aclweb.org/anthology/N15-1095

[23] C. Cortes, V. Kuznetsov, and M. Mohri, "Ensemble methods for structured prediction," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 1134–1142. [Online]. Available: http://jmlr.org/proceedings/papers/v32/cortesa14.html

[24] H. Baayen, R. Piepenbrock, and L. Gulikers, "The CELEX2 lexical database. ldc96l14," 1995.

[25] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th International Conf. on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 2001, pp. 282–289. [Online]. Available: citeseer.ist.psu.edu/lafferty01conditional.html