A WFST-based Log-linear Framework for Speaking-style Transformation

Graham Neubig, Shinsuke Mori, Tatsuya Kawahara

Graduate School of Informatics, Kyoto University
Sakyo-ku, Kyoto 606-8501, Japan

Abstract

When attempting to make transcripts from automatic speech recognition results, disfluency deletion, transformation of colloquial expressions, and insertion of dropped words must be performed to ensure that the final product is clean transcript-style text. This paper introduces a system for the automatic transformation of the spoken word to transcript-style language that enables not only deletion of disfluencies, but also substitutions of colloquial expressions and insertion of dropped words. A number of potentially useful features are combined in a log-linear probabilistic framework, and the utility of each is examined. The system is implemented using weighted finite state transducers (WFSTs) to allow for easy combination of features and integration with other WFST-based systems. On evaluation, the best system achieved a 5.37% word error rate, a 5.49% absolute gain over a rule-based baseline and a 1.54% absolute gain over a simple noisy-channel model.

Index Terms: speaking style transformation, disfluency detection, weighted finite state transducers, log-linear model

1. Introduction

Conventional ASR systems have been designed to faithfully decode what is uttered V, given an acoustic observation X, and thus were formulated to find the hypothesis which maximizes the posterior probability $P(V|X)$. In spontaneous speech, however, this scheme does not necessarily lead to intelligible transcripts as there are a number of disfluent phenomena, redundant or colloquial expressions, and incomplete sentences.

When professional human stenographers transcribe speech, they correct these phenomena to improve transcript readability. An ideal transcription system should have the ability to do so as well. It has been shown that this sort of transformation of ASR results is useful not only for human readers, but also for language model adaptation [1, 4] and down-the-line applications such as statistical machine translation (SMT) [12].

In this context, a number of works have been conducted on the detection of fillers, false-starts, and repairs. These adopted approaches such as noisy-channel modeling [3], CRFs [6], and weighted finite state transducers [8] (WFSTs). However, in more formal settings such as public speeches, congressional meetings, or lectures given to large audiences, disfluencies such as false starts and repairs are less frequent. On the other hand, human stenographers often must perform correction of colloquial or ungrammatical expressions to make a formal, document-style text. This is particularly true of languages such as Japanese where the disconnect between spoken and written speech is large [13].

Here, we focus on the task of transformation of Japanese verbatim text to transcript-style text. Previously, Hori *et al* [4] demonstrated that non-probabilistic transformation rules can be useful for language model adaptation in a WFST-based speech recognition system. Shitaoka *et al* [13] showed that a noisy-channel machine translation model can be used to transform verbatim transcripts into clean transcripts.

We present a system that uses SMT techniques to handle deletions, insertions, and substitutions simultaneously. We utilize a log-linear framework to combine non-independent features that provide benefits over a simple noisy-channel based model. All features are implemented using weighted finite state transducers (WFSTs) to provide flexibility in adding or removing features, and to enable integration with other WFST-based systems. The model was trained and evaluated on a corpus of meetings of the National Diet (Congress) of Japan.

In the remainder of the paper, Section 2 describes the task of speaking style transformation in more detail. Section 3 gives a description of log-linear models and the training of parameters. Section 4 presents various features that may be useful to the task. Section 5 gives a brief overview of WFSTs and describes the implementation of the search and features in the WFST framework. Section 6 gives experimental results and an analysis of the effect of each feature. Section 7 concludes with possible future directions.

2. Task definition

One of the major driving forces behind the development of systems to transform verbatim transcripts into clean transcripts was the DARPA EARS program's meta-data extraction (MDE) task [7]. The MDE task, broadly speaking, handled the deletion of extraneous words and insertion of sentence boundaries. This is, as shown in Figure 1, generally all that is necessary to transform a native speaker's English into transcript form.

However, these models are a limited subclass of a broader class of possible speaking style transformation models. In this paper, we focus on another speaking-style transformation task that requires not only deletions, but also insertions and substitutions: transcription of spontaneous Japanese. We broadly divide the corrections that are required to transform spoken Japanese into written Japanese into four categories:

- **Filler Deletions:** Japanese contains a number of words that are used exclusively, or nearly exclusively as fillers, and must be deleted.
- **Non-filler Deletions:** Other than fillers, there are a number of deletions that must be performed. These include repeats, repairs, and words that are only treated as fillers in certain contexts, such as ”*desu ne*,” which is used both as a filler and an actual word meaning ”it is, isn't it?”
- **Substitutions:** The majority of substitutions are colloquial expressions. The large disconnect between spoken Japanese and written Japanese makes it necessary to transform these colloquial expressions into formal expressions when creating transcripts.

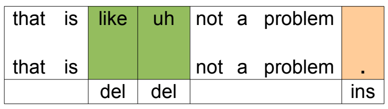

Figure 1: *An English sentence requiring only word deletions and punctuation insertion*

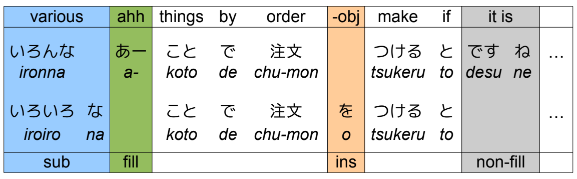
various ahh things by order -obj make if it is
いろんな あー こと で 注文 つける と です ね …
ironna a- koto de chu-mon tsukeru to desu ne
いろいろ な こと で 注文 を つける と …
iroiro na koto de chu-mon o tsukeru to
sub fill ins non-fill

Figure 2: *A Japanese phrase meaning "If you make orders for various things..." requiring four kinds of corrections*

- **Insertions:** Japanese spoken language allows the omitting of words such as particles (preposition-like function words). However, this is not permitted in written language, so omitted particles must be restored in a transcript.

The proposed system was trained and tested on a corpus of committee meetings of the National Diet (Congress) of Japan. A large corpus of 3.62M sentences of official Diet transcripts from 1999-2007 was used to train a language model. A smaller set (56.2K sentences) of aligned verbatim and clean transcripts was used to train translation probabilities. Parameter tuning was done on a 974 sentence set of held-out data.

Table 1: *Test set details*

Sentences			7,181
Words	Spoken		341,400
	Transcript		299,216
Deletions	Fillers	26,663	(47.3%)
	Non-fillers	20,143	(35.7%)
Substitutions		4,936	(8.8%)
Insertions		4,622	(8.2%)
Word Error %			16.40%

Testing was done on a set of meetings from 2007 that occured after all those in the training data. Statistics on the test corpus are shown in Table 1. It should be noted that while filler deletions are the most frequent correction at 47.3%, non-filler deletions, substitutions, and insertions account for 52.7% of the required corrections. This clearly indicates the necessity for an intelligent model to transform Japanese verbatim text into transcript-style text.

3. Noisy-channel and log-linear models

3.1. Noisy-channel modeling

In statistical models for applications such as ASR and statistical machine translation (SMT), the problem is defined as finding the most likely output given the input. For the transformation from spoken to transcript style, the input is a verbatim sentence V, and the output is a transcript-style sentence W. Thus, the task is to find the transcript-style sentence $\hat{W}$, with the maximal posterior probability $P(W|V)$

$$\hat{W} = \underset{W}{\mathrm{argmax}} P(W|V). \tag{1}$$

According to Bayes' law, the following maximization is equivalent to Equation (1):

$$\hat{W} = \underset{W}{\mathrm{argmax}} P(V|W)P(W). \tag{2}$$

This is often referred to as the *noisy-channel* approach. It is popular in the ASR and SMT communities because it allows the division of the modeling into two separate tasks. For speaking style transformation, the *translation model* probability $P(V|W)$ must be trained on a parallel corpus of verbatim and transcript-style sentences, but the *language model* probability $P(W)$ can be trained using a much larger set of transcript-style sentences with no verbatim counterparts.

Previous works such as Honal *et al* [3] and Shitaoka *et al* [13] have used this noisy channel model to correct deletions, substitutions, and insertions and create clean transcriptions.

3.2. Log-linear modeling

An alternative approach to noisy-channel modeling that has proven useful in SMT is the log-linear model[11]. A simple log-linear model is achieved by taking the logarithm of the noisy-channel model, and multiplying each log probability by a weight as shown in equation (3)

$$\hat{W} = \underset{W}{\mathrm{argmax}} \left[\alpha \log(P(V|W)) + \beta \log(P(W))\right]. \tag{3}$$

In the case that α and β are both equal to 1, the maximization of Equation (3) is equivalent to the maximization of Equation 2. This implies that log-linear models of this form are more general than simple noisy channel models, and thus can be expected to perform as well or better as long as the weights are chosen appropriately.

In addition, unlike normal noisy-channel models, log-linear models need make no assumptions about the independence of features, allowing the addition of an arbitrary number of possibly non-independent additional features [11].

$$\hat{W} = \underset{W}{\mathrm{argmax}} \left[\alpha f_1(V,W) + \beta f_2(V,W) + \ldots\right].$$

3.3. Weight training

When using log-linear models with non-independent features, it is important to train the weights of the various features used in the model. The method we chose here is minimum error rate training (MERT), which attempts to find values that minimize some error measure over held-out data [10]. Weights were trained to minimize WER over the held-out data set using the MERT tool included in open source SMT software Moses [5].

4. Feature selection

As mentioned in the previous section, one of the benefits of log-linear models is that they allow for the easy combination of non-independent features. This section gives a summary of the various features that we attempted and the motivation for using them.

4.1. Initial setup

The initial setup is a noisy-channel model with only the language model and translation model log probabilities as features.

$P(W)$ is modeled using a standard 3-gram language model smoothed using the Kneser-Ney method. $P(V|W)$ is modeled by assuming that each separate word transformation ($v_i \rightarrow w_i$) is independent, and taking the product of the probabilities.

$$\begin{aligned} P(V|W) &= \prod_i P(v_i|w_i) \\ &= \prod_i c(v_i, w_i)/c(w_i). \end{aligned} \quad (4)$$

The probability is calculated using the number of times v_i was translated into w_i and the total number of times w_i occurs. For insertions and deletions the empty string ϵ is aligned with the input and output respectively, and $P(\epsilon|w_i)$ and $P(v_i|\epsilon)$ are estimated. When a single word was aligned with multiple words, the multiple-word string was treated as one word.

4.2. Filler dictionary

The first additional feature was motivated by the fact that many of the deleted words were common fillers. A 23-word filler list was created, and a fixed score was added every time one of the fillers was deleted. This feature is expected to make the system work as well or better than the simple filler-removal baseline.

4.3. Transformation group penalty

Because there is a tendency for transformed areas to appear in groups (strings of deleted fillers, etc.), it makes sense to add a penalty that contributes to the cohesion of these groups. By adding a fixed penalty for each group of words translated, we can assure that in a phrase like "I uh don't um do," instead of creating two translation groups by deleting the underlined words in "I $\underline{\text{uh}}$ don't $\underline{\text{um}}$ do," the system will prefer to delete a single block as in "I $\underline{\text{uh don't um}}$ do."

4.4. Transformation type penalty

There is a general trend that some types of errors are more difficult than others to correct. In particular, deletions tend to be easier than substitutions. In order to take this fact into account, we introduce a separate penalty for each type of correction (deletion, substitution, and insertion), allowing the weight training algorithim to find an appropriate balance between precision and recall for each type.

4.5. Decomposed translation model

As in equation (4), the word transformation probability $P(v|w)$ is computed from the frequencies $c(v, w)$ and $c(w)$. By treating these two as separate elements it is possible to provide a separate weight for each. If $c(v, w)$ is assigned a higher weight than $c(w)$ by the weight training process, more frequently-observed transformations will be given precedence over less common transformations, which can be expected to increase the precision.

5. WFST implementation

We implemented the transformation model in the *weighted finite state transducer* (WFST) framework [9]. Finite state transducers are finite automata with transitions labeled with input and output symbols. WFSTs also assign a weight to transitions, allowing for the definition of weighted relations between two strings. In this case we use these weights to represent log probabilities.

Table 2: *The weights for each feature: language model (LM), $c(v, w)$ (CWV), $c(w)$ (CW), transformation groups (TG), deletions (D), substitutions (S), insertions (I), and fillers (F)*

LM	CWV	CW	TG	D	S	I	F
1.00	1.76	-1.00	1.34	0.00	0.26	-2.67	-0.46

Efficient algorithms for operations such as composition ($X \circ Y$), intersection ($X \cap Y$), determinization ($det(X)$), and minimization ($min(X)$) over WFSTs exist, allowing for creation of a single more complex model out of multiple WFSTs representing simpler models. In the proposed system, all features are implemented as single WFSTs and combined using composition or intersection.

Furthermore, after a single speaking-style transformation WFST SST that transforms spoken word V to transcript-style text W has been created, it can be seamlessly integrated with other WFST-based models. By composing with an ASR model that transforms acoustic observations X into verbatim text V, a model $ASR \circ SST$ that directly transforms acoustic observations into transcript-style text can be created. Conversely, composing SST with a language model LM that has been trained on transcript-style text results in a speaking-style adapted language model $SST \circ LM$.

The decoder used for the experiments was written using the open source toolkit OpenFst [2]. Small modifications were made to allow for the storing of multiple feature scores for later use in MERT, and beam-search style trimming to limit the search space.

6. Results and evaluation

6.1. Baseline

Experiments were performed on the data described in Section 2. A simple rule-based method that deletes all words occurring in the list of 23 common fillers was used as a baseline. The baseline achieved an 89.90% precision and 100% recall on filler deletions and performed no other edits. The WER after filler deletions was 10.86%, compared with 16.40% when no deletions were performed.

6.2. Results

The model that performed best on the test data set was the model that included all of the features listed in Section 4. It achieved a 5.37% error rate, which is a 5.49% absolute improvement over the filler-removed baseline.

The system was able to successfully perform 658 separate varieties of corrections, 328 of which occurred in the test corpus only once. This demonstrates a clear advantage of the proposed system over a rule-based system, which would require handcrafted rules for each type of correction.

In addition, the proposed model achieved a 1.54% absolute improvement over the simple noisy-channel model with equal translation model and language model weights. This is due to the fact that the proposed system made, in general, fewer but more accurate corrections than the noisy channel model. In particular, the proposed system did much better at inserting dropped particles and determining whether non-filler deletion

Table 3: *Performance of the model with all features, the effect of removing each feature, the noisy channel model, and the baseline*

		Fillers [%]		Other Deletions [%]		Substitutions [%]		Insertions [%]	
	WER [%]	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Recall
All Features Used	5.37	99.12	97.58	85.02	52.03	88.53	56.00	87.47	54.39
-Filler Dictionary	5.46	99.22	97.49	79.85	54.68	84.43	57.80	82.47	56.10
-Type Penalty	5.62	98.83	97.95	83.88	51.05	85.46	57.52	85.97	44.81
-Translation Groups	5.67	98.94	96.69	79.68	52.65	88.13	54.32	82.06	56.82
-Decomposed Model	5.74	98.24	98.87	75.02	55.94	83.78	54.09	80.77	52.51
Noisy Channel Model	6.91	98.74	97.53	63.36	59.70	68.17	60.43	65.38	55.78
Filler-removed Baseline	10.86	89.90	100.0	-	-	-	-	-	-

candidates should actually be deleted or not.

6.3. Feature effects

The weights assigned to each feature in the model containing all the features are shown in Table 2. In order to examine which of the features were contributing to the gains in accuracy, we removed one feature at a time from the optimal model and examined the change in performance of each type of correction. The performance of the systems with one feature removed is shown in Table 3, with the filler-deleted baseline and the unweighted noisy-channel models provided for reference. The difference between the best model and all other models (excluding the model with only the filler dictionary removed) is statistically significantly according to the two-proportions z-test with a confidence level of 1%.

The feature that contributed the most to the gain in accuracy was the decomposed translation model. By increasing the weight on the joint frequency $c(v, w)$, frequent transformations were given precedence over uncommon transformations. This resulted in a large gain in precision across the board at the cost of a small drop in recall.

The translation group penalty feature also had a relatively large effect on accuracy. This was the feature that contributed most to determining whether words that had the potential to be fillers should actually be deleted or not. It also had an effect of preventing borderline transformations, increasing the precision and reducing the recall in all four transformation categories.

The transformation type penalty also made a significant contribution. It added a large negative penalty (bonus) to insertions, increasing recall by nearly 10 points, and increased precision of substitutions at the cost of recall.

The filler dictionary showed a small (statistically insignificant) gain, slightly increasing the recall of both filler and non-filler deletions and raising precision in the other two transformation types.

7. Conclusion and future directions

The results of the evaluation demonstrate that using probabilistic methods have clear advantages over simple deletion of fillers when transforming verbatim text into transcript-style text. Proper selection and weight training of additional features leads to further increases in transformation accuracy. Because the described features are implemented in the WFST framework, the translation model can be incorporated directly with WFST-based ASR systems or be used for language model adaptation.

The ultimate goal of the proposed system must be to function on ASR output. ASR output presents a new set of challenges, including the insertion of punctuation and handling of incorrectly recognized words.

8. Acknowledgements

The authors would like to kindly thank Markus Dreyer for providing access to his code implementing expectation semi-rings in OpenFst.

9. References

[1] Y. Akita and T. Kawahara. Topic-independent speaking-style transformation of language model for spontaneous speech recognition. In *Proc. ICASSP2007*, pages 33–36, 2007.

[2] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri. OpenFst: a general and efficient weighted finite-state transducer library. In *Proc. CIAA '07*, pages 11–23, 2007.

[3] M. Honal and T. Schultz. Correction of disfluencies in spontaneous speech using a noisy-channel approach. In *Proc. EuroSpeech2003*, pages 2781–2784, Geneva, Switzerland, 2003.

[4] T. Hori, D. Willett, and Y. Minami. Language model adaptation using wfst-based speaking-style translation. In *Proc. ICASSP2003*, pages 228–231, 2003.

[5] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. ACL07*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[6] Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Transactions on Audio, Speech, and Language Processing*, pages 1526–1540, 2006.

[7] Y. Liu, E. Shriberg, A. Stolcke, B. Peskin, J. Ang, D. Hillard, M. Ostendorf, M. Tomalin, P. Woodland, and M. Harper. Structural metadata research in the EARS program. In *Proc. ICASSP2005*, pages 957–960, 2005.

[8] S. Maskey, B. Zhou, and Y. Gao. A phrase-level machine translation approach for disfluency detection using weighted finite state transducers. In *Proc. InterSpeech2006*, pages 749–752, 2006.

[9] M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.

[10] F. J. Och. Minimum error rate training in statistical machine translation. In *Proc. ACL03*, pages 160–167, 2003.

[11] F. J. Och and H. Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. ACL02*, pages 295–302, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[12] S. Rao, I. Lane, and T. Schultz. Improving spoken language translation by automatic disfluency removal: Evidence from conversational speech transcripts. In *Machine Translation Summit XI*, pages 177–180, Copenhagen, Denmark, September 2007.

[13] K. Shitaoka, H. Nanjo, and T. Kawahara. Automatic transformation of lecture transcription into document style using statistical framework. In *Proc. InterSpeech2004*, pages 2169–2172, 2004.