Two Approaches to Correcting Homophone Confusions in a Hybrid Machine Translation System

Pierrette Bouillon[1], **Johanna Gerlach**[1], **Ulrich Germann**[2], **Barry Haddow**[2], **Manny Rayner**[1]
(1) FTI/TIM, University of Geneva, Switzerland
{Pierrette.Bouillon,Johanna.Gerlach,Emmanuel.Rayner}@unige.ch
(2) School of Informatics, University of Edinburgh, Scotland
{ugermann,bhaddow}@inf.ed.ac.uk

Abstract

In the context of a hybrid French-to-English SMT system for translating online forum posts, we present two methods for addressing the common problem of homophone confusions in colloquial written language. The first is based on hand-coded rules; the second on weighted graphs derived from a large-scale pronunciation resource, with weights trained from a small bicorpus of domain language. With automatic evaluation, the weighted graph method yields an improvement of about $+0.63$ BLEU points, while the rule-based method scores about the same as the baseline. On contrastive manual evaluation, both methods give highly significant improvements ($p < 0.0001$) and score about equally when compared against each other.

1 Introduction and motivation

The data used to train Statistical Machine Translation (SMT) systems is most often taken from the proceedings of large multilingual organisations, the generic example being the Europarl corpus (Koehn, 2005); for academic evaluation exercises, the test data may well also be taken from the same source. Texts of this kind are carefully cleaned-up formal language. However, real MT systems often need to handle text from very different genres, which as usual causes problems.

This paper addresses a problem common in domains containing informally written text: spelling errors based on homophone confusions. Concretely, the work reported was carried out in the context of the ACCEPT project, which deals with the increasingly important topic of translating online forum posts; the experiments we describe were performed using French data taken from the Symantec forum, the concrete task being to translate it into English. The language in these posts is very far from that which appears in Hansard. People write quickly and carelessly, and no attempt is made to clean up the results. In particular, spelling is often uncertain.

One of the particular challenges in the task considered here is that French has a high frequency of homophones, which often cause confusion in written language. Everyone who speaks English is familiar with the fact that careless writers may confuse *its* (“of or belonging to it”) and *it’s* (contraction of “it is” or “it has”). French has the same problem, but to a much greater degree. Even when someone is working in an environment where an online spell-checker is available, it is easy to write *ou* (“or”) instead of *où* (“where”), *la* (“the-feminine”) instead of *là* (“there”) or *ce* (“this”) instead of *se* (“him/herself”). Even worse, there is systematic homophony in verb-form endings: for example, *utiliser* (“to use”) *utilisez* (“you use”) and *utilisé* (“used”) are all homophones.

In French posts from the Symantec forum, we find that between 10% and 15% of all sentences contain at least one homophone error, depending on exactly how the term is defined[1]. Substituting a word with an incorrect homophone will often result in a translation error. Figure 1 shows typical examples of homophone errors and their effect on translation.

The core translation engine in our application is a normal SMT system, bracketed between pre- and post-editing phases. In what follows, we contrast two different approaches to handling homophone errors, which involve pre-editing in different ways. The first approach is based on knowledge-intensive construction of regular expression rules, which use the surrounding context to correct the most frequent types of homophone

[1] Unclear cases include hyphenation, elison and some examples of missing or incorrect accents.

Proceedings of the Second Workshop on Hybrid Approaches to Translation, pages 109–116,
Sofia, Bulgaria, August 8, 2013. ©2013 Association for Computational Linguistics

	source	automatic translation
original	**La sa** ne pose pas de problème ...	**The its** is not the issue ...
corrected	**Là ça** ne pose pas de problème ...	**Here it** is not a problem
original	... (du moins on ne **recoit** pas l'alerte).	... (at least **we do not recoit** alert).
corrected	... (du moins on ne **reçoit** pas l'alerte).	.. (at least **it does not receive** the alert).

Figure 1: Examples of homophone errors in French forum data, contrasting English translations produced by the SMT engine from plain and corrected versions.

confusions.

The second is an engineering method: we use a commercial pronunciation-generation tool to generate a homophone dictionary, then use this dictionary to turn the input into a weighted graph where each word is replaced by a weighted disjunction of homophones. Related, though less elaborate, work has been reported by Bertoldi et al. (2010), who address spelling errors using a character-level confusion network based on common character confusions in typed English and test them on artificially created noisy data. Formiga and Fonollosa (2012) also used character-based models to correct spelling on informally written English data.

The two approaches in the present paper exploit fundamentally different knowledge sources in trying to identify and correct homophone errors. The rule-based method relies exclusively on source-side information, encoding patterns indicative of common French homophone confusions. The weighted graph method shifts the balance to the target side; the choice between potential homophone alternatives is made primarily by the target language model, though the source language weights and the translation model are also involved.

The rest of the paper is organised as follows. Section 2 describes the basic framework in more detail, and Section 3 the experiments. Section 4 summarises and concludes.

2 Basic framework

The goal of the ACCEPT project is to provide easy cross-lingual access to posts in online forums. Given the large variety of possible technical topics and the limited supply of online gurus, it frequently happens that users, searching forum posts online, find that the answer they need is in a language they do not know.

Currently available tools, for example Google Translate, are of course a great deal better than nothing, but still leave much to be desired. When one considers that advice given in an online forum may not be easy to follow even for native language speakers, it is unsurprising that a Google-translated version often fails to be useful. There is consequently strong motivation to develop an infrastructure explicitly designed to produce high-quality translations. ACCEPT intends to achieve this by a combination of three technologies: pre-editing of the source; domain-tuned SMT; and post-editing of the target. The pre- and post-editing stages are performed partly using automatic tools, and partly by manual intervention on the part of the user communities which typically grow up around online forums. We now briefly describe the automatic parts of the system.

2.1 SMT engine and corpus data

The SMT engine used is a phrase-based system trained with the standard *Moses* pipeline (Koehn et al., 2007), using GIZA++ (Och and Ney, 2000) for word alignment and SRILM (Stolcke, 2002) for the estimation of 5-gram Kneser-Ney smoothed (Kneser and Ney, 1995) language models.

For training the translation and lexicalised reordering models we used the releases of europarl and news-commentary provided for the WMT12 shared task (Callison-Burch et al., 2012), together with a dataset from the ACCEPT project consisting mainly of technical product manuals and marketing materials.

For language modelling we used the target sides of all the parallel data, together with approximately 900 000 words of monolingual English data extracted from web forums of the type that we wish to translate. Separate language models were trained on each of the data sets, then these were linearly interpolated using SRILM to minimise perplexity on a heldout portion of the forum data.

For tuning and testing, we extracted 1022 sentences randomly from a collection of monolingual French Symantec forum data (distinct from the monolingual English forum data), translated these using Google Translate, then post-edited to create references. The post-editing was performed by a native English speaker, who is also fluent in French. This 1022-sentence parallel text was then split into two equal halves (`devtest_a` and `devtest_b`) for minimum error rate tuning (MERT) and testing, respectively.

2.2 Rule-based pre-editing engine

Rule-based processing is carried out using the Acrolinx engine (Bredenkamp et al., 2000), which supports spelling, grammar, style and terminology checking. These methods of pre-editing were originally designed to be applied by authors during the technical documentation authoring process. The author gets error markings and improvement suggestions, and decides about reformulations. It is also possible to apply the provided suggestions automatically as direct reformulations. Rules are written in a regular-expression-based formalism which can access tagger-generated part-of-speech information. The rule-writer can specify both positive evidence (patterns that will trigger application of the rule) and negative evidence (patterns that will block application).

3 Experiments

We compared the rule-based and weighted graph approaches, evaluating each of them on the 511 sentence `devtest_b` corpus. The baseline SMT system, with no pre-editing, achieves an average BLEU score of 42.47 on this set.

3.1 The rule-based approach

Under the ACCEPT project, a set of lightweight pre-editing rules have been developed specifically for the Symantec Forum translation task. Some of the rules are automatic (direct reformulations); others present the user with a set of suggestions. The evaluations described in Gerlach et al. (2013) demonstrate that pre-editing with the rules has a significant positive effect on the quality of SMT-based translation.

The implemented rules address four main phenomena: differences between informal and formal language (Rayner et al., 2012), differences between local French and English word-order, elision/punctuation, and word confusions. Rules for resolving homophone confusions belong to the fourth group. They are shown in Table 1, together with approximate frequencies of occurrence in the development corpus.

Table 1: Hand-coded rules for homophone confusions and per-sentence frequency of applicability in the development corpus. Some of the rules also cover non-homophone errors, so the frequency figures are slight overestimates as far as homophones are concerned.

Rule	Freq.
a/as/à	4.17%
noun phrase agreement	3.20%
incorrect verb ending (er/é/ez)	2.90%
missing hyphenation	2.08%
subject verb agreement	1.90%
missing elision	1.26%
du/dû	0.35%
la/là	0.32%
ou/où	0.28%
ce/se	0.27%
Verb/noun	0.23%
tous/tout	0.22%
indicative/imperative	0.19%
future/conditional tense	0.14%
sur/sûr	0.10%
quel que/quelque	0.08%
ma/m’a	0.06%
quelle/qu’elle/quel/quels	0.05%
ça/sa	0.04%
des/dès	0.04%
et/est	0.02%
ci/si	0.01%
m’y/mi/mis	0.01%
other	0.17%
Total	18.09%

The set of Acrolinx pre-editing rules potentially relevant to resolution of homophone errors was applied to the `devtest_b` set test corpus (Section 2.1). In order to be able to make a fair comparison with the weighted-graph method, we only used rules with a unique suggestion, which could be run automatically. Applying these rules produced 430 changed words in the test corpus, but did not change the average BLEU score significantly (42.38).

Corrections made with a human in the loop, used as “oracle” input for the SMT system, by the way, achieve an average BLEU score[2] of 43.11 — roughly on par with the weighted-graph approach described below.

3.2 The weighted graph approach

In our second approach, the basic idea is to transform the input sentence into a *confusion network* (Bertoldi et al., 2008) which presents the translation system with a weighted list of homophone alternatives for each input word. The system is free to choose a path through a network of words that optimizes the internal hypothesis score; the weighting scheme for the alternatives can be used to guide the decoder. The conjecture is that the combination of the confusion network weights, the translation model and the target language model can resolve homophone confusions.

3.2.1 Defining sets of confusable words

To compile lists of homophones, we used the commercial Nuance Toolkit `pronounce` utility as our source of French pronunciation information.

We began by extracting a list of all the lexical items which occurred in the training portion of the French Symantec forum data, giving us 30 565 words. We then ran `pronounce` over this list. The Nuance utility does not simply perform table lookups, but is capable of creating pronunciations on the fly; it could in particular assign plausible pronunciations to most of the misspellings that occurred in the corpus. In general, a word is given more than one possible pronunciation. This can be for several reasons; in particular, some sounds in French can systematically be pronounced in more than one way, and pronunciation is often also dependent on whether the word is followed by a consonant or vowel. Table 2 shows examples.

Using the data taken from `pronounce`, we grouped words together into clusters which have a common pronunciation; since words typically have more than one pronunciation, they will typically also belong to more than one cluster. We then contructed sets of possible alternatives for words by including, for each word W, all the words W' such that W and W' occurred in the same cluster; since careless French writing is also characterised by mistakes in placing accents, we added all words W' such that W and W' are identical up to dropping accents. Table 3 shows typical results.

Word	Pronunciation
ans	A˜ A˜z
prévu	p r E v y p r e v y
québec	k e b E k
roule	r u l r u l *

Table 2: Examples of French pronunciations generated by `pronounce`. The format used is the Nuance version of ARPABET.

Intuitively, it is in general unlikely that, on seeing a word which occurs frequently in the corpus, we will want to hypothesize that it may be a misspelling of one which occurs very infrequently. We consequently filtered the sets of alternatives to remove all words on the right whose frequency was less than 0.05 times that of the word on the left.

Table 3: Examples of sets of possible alternatives for words, generated by considering both homophone and accent confusions.

Word	Alternatives
aux	au aux haut
créer	créer créez créé créée créées créés
côte	cote coté côte côté quot quote
hôte	haut haute hôte hôtes
il	e elle elles il ils l le y
mène	main mené mène
nom	nom noms non
ou	ou où
saine	sain saine saines scène seine
traits	trait traits tray tre tres très

3.2.2 Setting confusion network weights

In a small series of preliminary experiments we first tested three naïve weighting schemes for the confusion networks.

- using a **uniform** distribution that assigns equal weight to all spelling alternatives;
- setting weights proportional to the **unigram probability** of the word in question;
- computing the weights as state probabilities in a trellis with the **forward-backward** algorithm (Rabiner, 1989), an algorithm widely

[2] With parameter sets from tuning the system on raw input and input preprocessed with the fully automatic rules; cf. Sec. 3.3.

Table 4: Decoder performance with different confusion network weighting schemes.

weighting scheme	av. BLEU[a]	std.
none (baseline system)	42.47	± .22
uniform	41.50	± .37
unigram	41.58	± .26
fwd-bwd (bigram)	41.81	± .16
bigram context (interpolated)	43.10	± .32

[a] Based on muliple tuning runs with random parameter initializations.

> used in speech recognition. Suppose that each word $\hat{w}_i$ in the observed translation input sentence is produced while the writer has a particular "true" word $w_i \in C_i$ in mind, where C_i is the set of words confusable with $\hat{w}_i$. For the sake of simplicity, we assume that within a confusion set, all "true word" options are equally likely, i.e., $p(\hat{w}_i \,|\, w_i = x) = \frac{1}{|C_i|}$ for $x \in C_i$. The writer chooses the next word w_{i+1} according to the conditional word bigram probability $p(w_{i+1} \,|\, w_i)$.
>
> The *forward* probability $fwd_i(x)$ is the probability of arriving in state $w_i = x$ at time i, regardless of the sequence of states visited en-route; the backward probability $bwd_i(x)$ is the probability of arriving at the end of the sentence coming from state $w_i = x$, regardless of the path taken. These probabilities can be computed efficiently with dynamic programming.
>
> The weight assigned to a particular homophone alternative x at position i in the confusion network is the joint forward and backward probability:
>
> $$weight_i(x) = fwd_i(x) \cdot bwd_i(x).$$

In practice, it turns out that these three naïve weighting schemes do more harm than good, as the results in Table 4 show. Clearly, they rely too much on overall language statistics (unigram and bigram probabilities) and pay too little attention to the actual input.

We therefore designed a fourth weighting scheme ("**bigram context interpolated**") that gives more weight to the observed input and computes the weights as the average of two score components. The first is a binary feature function that assigns 1 to each word actually observed in the input, and 0 to its homophone alternatives. The second component is the bigram-based in-context probability of each candidate. Unlike the forward-backward weighting scheme, which considers all possible context words for each candidate (as specified in the respective confusion sets), the new scheme only considers the words in the actual input as context words.

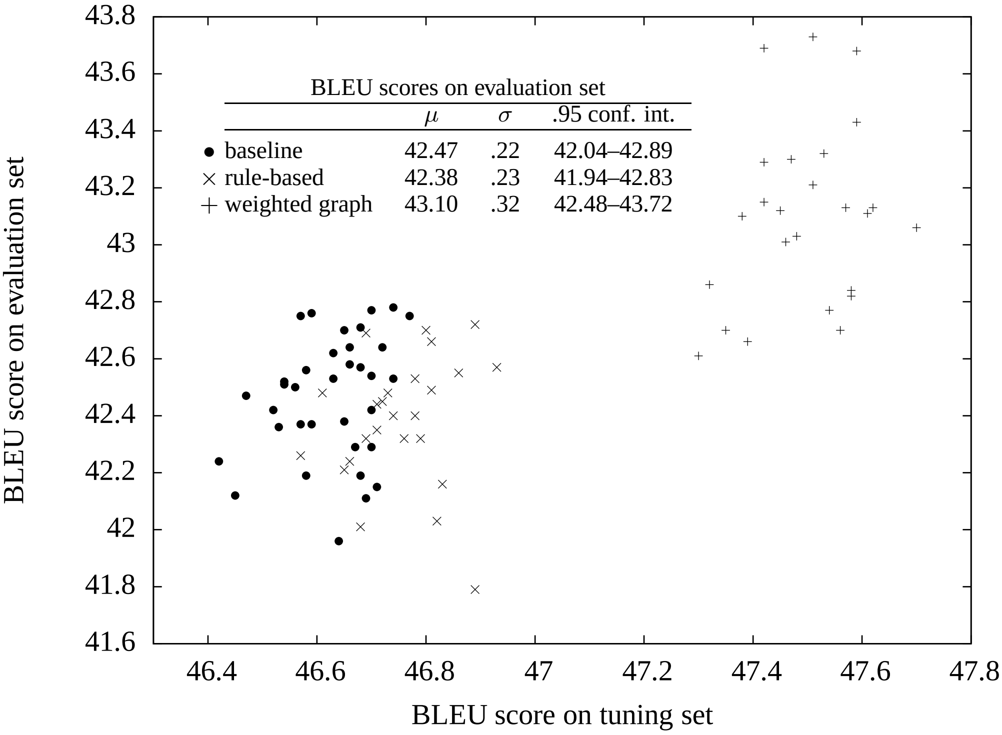
It would have be desirable to keep the two score components separate and tune their weights together with all the other parameters of the SMT system. Unfortunately, the current implementation of confusion network-based decoding in the *Moses* decoder allows only one single weight in the specification of confusion networks, so that we had to combine the two components into one score before feeding the confusion network into the decoder.

With the improved weighting scheme, the confusion network approach does outperform the baseline system, giving an average BLEU of 43.10 ($+0.63$).

3.3 Automatic evaluation (BLEU)

Due to the relatively small size of the evaluation set and instability inherent in minimum error rate training (Foster and Kuhn, 2009; Clark et al., 2011), results of *individual* tuning and evaluation runs can be unreliable. We therefore preformed multiple tuning and evaluation runs for each system (baseline, rule-based and weighted graph). To illustrate the precision of the BLEU score on our data sets, we plot in Fig. 2 for each individual tuning run the BLEU score achieved on the tuning set (x-axis) against the performance on the evaluation set (y-axis). The variance along the x-axis for each system is due to search errors in parameter optimization. Since the search space is not convex, the tuning process can get stuck in local maxima. The apparent poor local correlation between performance on the tuning set and performance on the evaluation set for each system shows the effect of the sampling error.

With larger tuning and evaluation sets, we would expect the correlation between the two to improve. The scatter plot suggests that the weighted-graph system does on average produce significantly better translations (with respect to BLEU) than both the baseline and the rule-based system, whereas the difference between the baseline and the rule-based system is within the range



BLEU scores on evaluation set

	μ	σ	.95 conf. int.
● baseline	42.47	.22	42.04–42.89
× rule-based	42.38	.23	41.94–42.83
+ weighted graph	43.10	.32	42.48–43.72

Figure 2: BLEU scores (in points) for the baseline, rule-based and weighted graph-based systems.

of statistical error.

To study the effect of tuning condition (tuning on raw vs. input pre-processed by rules), we also translated both the raw and the pre-processed evaluation corpus with all parameter setting that we had obtained during the various experiments. Figure 3 plots (with solid markers) performance on raw input (x-axis) against translation of pre-processed input (y-axis). We observe that while preprocessing harms performance for certain parameter settings, most of the time proprocessing does lead to improvements in BLEU score. The slight deterioration we observed when comparing system tuned on exactly the type of input that they were to translate later (i.e., raw or preprocessed) seems to be a imprecision in the measurement caused by training instability and sampling error rather than the result of systematic input deterioration due to preprocessing. Overall, the improvements are small and not statistically significant, but there appears to be a positive trend.

To gauge the benefits of more extensive preprocessing and input error correction we produced and translated 'oracle' input by also applying rules from the Acrolinx engine that currently require a human in the loop who decides whether or not the rule in question should be applied. The boost in performance is shown by the hollow markers in Fig. 3. Here, translation of pre-processed input consistently fares better than translation of the raw input.

3.4 Human evaluation

Although BLEU suggests that the weighted-graph method significantly outscores both the baseline and the rule-based method ($p < 0.05$ over 25 tuning runs), the absolute differences are small, and we decided that it would be prudent to carry out a human evaluation as well. Following the methodology of Rayner et al. (2012), we performed contrastive judging on the Amazon Mechanical Turk (AMT) to compare different versions of the system. Subjects were recruited from Canada, a bilingual French/English country, requesting English native speakers with good written French; we also limited the call to AMT workers who had already completed at least 50 assignments, at least 80% of which had been accepted. Judging assignments were split into groups of 20 triplets, where each triplet consisted of a source sentence and two different target sentences; the judge was asked to say which translation was better, using a five-point scale {better, slightly-better, about-equal, slightly-worse, worse}. The order of the two targets was

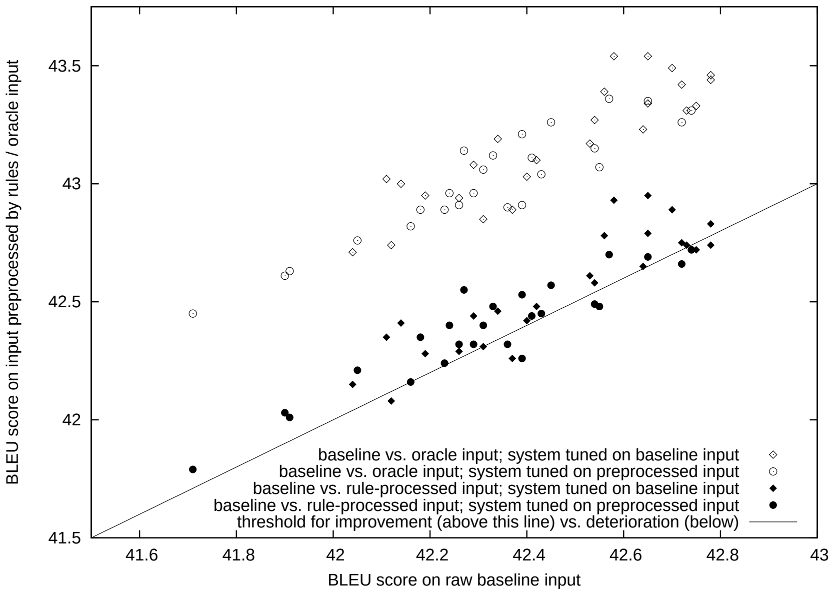


Figure 3: BLEU scores (in points) the two input conditions "baseline" and "rule-based" (solid markers). The hollow markers show the BLEU score on human-corrected 'oracle' input using a more extensive set of rules / suggestions from the Acrolinx engine that require a human in the loop.

randomised. Judges were paid $1 for each group of 20 triplets. Each triplet was judged three times.

Using the above method, we posted AMT tasks to compare a) the baseline system against the rule-based system, b) the baseline system against the best weighted-graph system (**interpolated-bigram**) from Section 3.2.2 and c) the rule-based system and the weighted-graph system against each other. The results are shown in Table 5; in the second and third columns, disagreements are resolved by majority voting, and in the fourth and fifth we only count cases where the judges are unanimous, the others being scored as unclear. In both cases, we reduce the original five-point scale to a three-point scale {better, equal/unclear, worse}[3]. Irrespective of the method used to resolve disagreements, the differences "rule-based system/baseline" and "weighted-graph system/baseline" are highly significant ($p < 0.0001$) according to the McNemar sign test, while the difference "rule-based system/weighted-graph system" is not significant.

Table 5: Comparison between baseline, rule-based and weighted-graph versions, evaluated on the 511-utterance devtest_b corpus and judged by three AMT-recruited judges. Figures are presented both for majority voting and for unanimous decisions only.

	Majority		Unanimous	
baseline vs rule-based				
baseline better	83	16.2%	48	9.4%
r-based better	204	40.0%	161	31.5%
Unclear	36	7.0%	93	18.1%
Equal	188	36.8%	209	40.9%
baseline vs weighted-graph				
baseline better	115	22.5%	52	10.1%
w-graph better	193	37.8%	119	23.3%
Unclear	46	9.0%	99	19.4%
Equal	157	30.7%	241	47.2%
rule-based vs weighted-graph				
r-based better	141	27.6%	68	13.3%
w-graph better	123	24.1%	70	13.7%
Unclear	25	4.9%	142	27.8%
Equal	222	43.4%	231	45.2%

We were somewhat puzzled that BLEU makes the weighted-graph system clearly better than the rule-based one, while manual evaluation rates them as approximately equal. The explanation seems to be to do with the fact that manual evaluation operates at the sentence level, giving equal importance to all sentences, while BLEU oper-

[3] For reasons we do not fully understand, we get better inter-judge agreement this way than we do when we originally ask for judgements on a three-point scale.

ates at the word level and consequently counts longer sentences as more important. If we calculate BLEU on a per-sentence basis and then average the scores, we find that the results for the two systems are nearly the same; per-sentence BLEU differences also correlate reasonably well with majority judgements (Pearson correlation coefficient of 0.39). It is unclear to us, however, whether the difference between per-sentence and per-word BLEU evaluation points to anything particularly interesting.

4 Conclusions

We have presented two methods for addressing the common problem of homophone confusions in colloquial written language in the context of an SMT system. The weighted-graph method produced a small but significant increase in BLEU, while the rule-based one was about the same as the baseline. Both methods, however, gave clearly significant improvements on contrastive manual evaluation carried out through AMT, with no significant difference in performance when the two were compared directly.

The small but consistent improvements in BLEU score that we observed with the human-in-the-loop oracle input over the fully automatic rule-based setup invite further investigation. How many of the decisions currently left to the human can be automated? Is there a fair way of comparing and evaluating fully automatic against semi-automatic setups? Work on these topics is in preparation and will be reported elsewhere.

Acknowledgements

The work described in this paper was performed as part of the Seventh Framework Programme ACCEPT project, under grant agreement 288769.

References

Bertoldi, Nicola, Mauro Cettolo, and Marcello Federico. 2010. “Statistical machine translation of texts with misspelled words.” *NAACL*. Los Angeles, CA, USA.

Bertoldi, Nicola, Richard Zens, Marcello Federico, and Wade Shen. 2008. “Efficient speech translation through confusion network decoding.” *IEEE Transactions on Audio, Speech & Language Processing*, 16(8):1696–1705.

Bredenkamp, Andrew, Berthold Crysmann, and Mirela Petrea. 2000. “Looking for errors : A declarative formalism for resource-adaptive language checking.” *LREC*. Athens, Greece.

Callison-Burch, Chris, Philipp Koehn, Christof Monz, et al. (eds.). 2012. *Seventh Workshop on Statistical Machine Translation (WMT)*. Montréal, Canada.

Clark, Jonathan H., Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. “Better hypothesis testing for statistical machine translation: Controlling for optimizer instability.” *ACL-HLT*. Portland, OR, USA.

Formiga, Lluis and José A. R. Fonollosa. 2012. “Dealing with input noise in statistical machine translation.” *COLING*. Mumbai, India.

Foster, George and Roland Kuhn. 2009. “Stabilizing minimum error rate training.” *WMT*. Athens, Greece.

Gerlach, Johanna, Victoria Porro, Pierrette Bouillon, and Sabine Lehmann. 2013. “La pré-édition avec des règles peu coûteuses, utile pour la TA statistique?” *TALN-RECITAL*. Sables d’Olonne, France.

Kneser, Reinhard and Hermann Ney. 1995. “Improved backing-off for m-gram language modeling.” *ICASSP*. Detroit, MI, USA.

Koehn, Philipp. 2005. “Europarl: A parallel corpus for statistical machine translation.” *MT Summit X*. Phuket, Thailand.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, et al. 2007. “Moses: Open source toolkit for statistical machine translation.” *ACL Demonstration Session*. Prague, Czech Republic.

Och, Franz Josef and Hermann Ney. 2000. “Improved statistical alignment models.” *ACL*. Hong Kong.

Rabiner, Lawrence R. 1989. “A tutorial on hidden markov models and selected applications in speech recognition.” *Proceedings of the IEEE*, 257–286.

Rayner, Manny, Pierrette Bouillon, and Barry Haddow. 2012. “Using source-language transformations to address register mismatches in SMT.” *AMTA*. San Diego, CA, USA.

Stolcke, Andreas. 2002. “SRILM - an extensible language modeling toolkit.” *ICSLP*. Denver, CO, USA.