Unpacking and Transforming Feature Functions: New Ways to Smooth Phrase Tables

Boxing Chen, Roland Kuhn, George Foster, and Howard Johnson
National Research Council of Canada, Gatineau, Québec, Canada
First.Last@nrc.gc.ca

Abstract

State of the art phrase-based statistical machine translation systems typically contain two features which estimate the "forward" and "backward" conditional translation probabilities for a given pair of source and target phrase. These two "relative frequency" (RF) features are derived from three counts: the joint count of the source and target phrase and their marginal counts. We propose to "unpack" these three statistics, making them independent "3-count" features instead of two RF features. In our experiments, the 3-count features perform better than the RF ones in three of four systems we tested. By transforming and generalizing these 3-count features slightly, further improvements are obtained. Furthermore, under several different experimental conditions, we compare 3-count and generalized 3-count features to new features derived from Kneser-Ney smoothing, to a new low-frequency penalty feature, and to several known smoothing/discounting schemes. Generalized 3-count performs similarly to or better than all of the smoothing methods except modified Kneser-Ney. In our experiments, the best phrase table (**not** language model) smoothing yields +0.6-1.4 BLEU.

1 Introduction

The translation model component of state of the art phrase-based statistical machine translation (SMT) systems (the "phrase table") consists of conditional probabilities for phrase pairs observed in the training data. However, estimation of these probabilities is hindered by data sparseness; thus, phrase table smoothing techniques are often applied (Foster *et al.*, 2006). The choice of smoothing technique for phrase tables often has a larger impact on performance than other aspects of a statistical machine translation system.

This paper is originally motivated by the observation that there is something strange about the way in which the translation model (TM) component of phrase-based SMT systems is often computed. The two translation probabilty features are estimated with "relative frequency" (RF) formulas that employ three counts: the joint count of the source and target phrase, and the marginal counts for the source phrase and the target phrase. Why wouldn't we use the three statistics that yield relative frequency estimates as independent features instead of using them to derive two RF features? We will call the replacement of an SMT feature function with features consisting of the statistics from which it is derived the "unpacking" of the feature function.

The second motivation for this paper is the comparison of phrase table smoothing techniques. Apart from (Foster *et al.*, 2006), little has been published on discounting and smoothing applied to phrase tables. System descriptions for evaluations like WMT, NIST, or IWSLT often mention <u>one</u> smoothing technique that was used in a system, but not why it was chosen instead of others. In addition to the new techniques described in this paper, the invention of the low-frequency indicator features (Mauser *et al.*, 2007) postdated the (Foster *et al.*, 2006) paper; it is of interest to compare these new techniques to the ones in that paper. Furthermore, today's state of the art phrase-based systems, including the one that serves as the baseline in the experiments

described here, perform considerably better than the one in (Foster *et al.*, 2006); the conclusions in that paper might be almost irrelevant to current systems. Thus, we think it is time to carry out a new experimental comparison of a variety of smoothing techniques.

The remainder of this paper is organized as following. Section 2 will introduce some existing smoothing techniques. In section 3, we will unpack and transform the two RF feature functions and Kneser-Ney phrase table smoothing. Section 4 is the experiments and discussion. Section 5 ends the paper with conclusion and future work.

2 Existing Smoothing Techniques

The phrase table consists of conditional probabilities of co-occurrence for source-language phrases **s** and target-language phrases **t**. "Relative frequency" (RF) estimates for these probabilities are obtained from a phrase pair extraction procedure applied to a bilingual training corpus (Koehn *et al.*, 2003). Let c(**s**) be the count of a source phrase **s**, c(**t**) the count of a target phrase **t**, and c(**s**,**t**) the number of times **s** and **t** are aligned to form a phrase pair.

Relative frequency (RF) estimates of "forward" probability $P_{RF}(\mathbf{t}|\mathbf{s})$ and "backward" probability $P_{RF}(\mathbf{s}|\mathbf{t})$ are

$$P_{RF}(\underline{t} \mid \underline{s}) = \frac{c(\underline{s},\underline{t})}{c(\underline{s})} \quad (1)$$

$$P_{RF}(\underline{s} \mid \underline{t}) = \frac{c(\underline{s},\underline{t})}{c(\underline{t})} \quad (2)$$

These RF estimates are often combined with "lexical weighting" (LW) estimates of the same probabilities $P_{LW}(\mathbf{t}|\mathbf{s})$ and $P_{LW}(\mathbf{s}|\mathbf{t})$, based on co-occurrence counts of the individual words making up **s** and **t**. Thus, the TM score is typically of this form (Och and Ney, 2002):

$$S_{TM} = \lambda_1 \times log[P_{RF}(\underline{t}|\underline{s})] + \lambda_2 \times log[P_{RF}(\underline{s}|\underline{t})] + \lambda_3 \times log[P_{LW}(\underline{t}|\underline{s})] + \lambda_4 \times log[P_{LW}(\underline{s}|\underline{t})]. \quad (3)$$

The λ's are often estimated by the minimum error rate training (MERT) algorithm (Och, 2003).

For the following two, the implementation details are as in (Foster *et al.*, 2006).

Good-Turing: observed counts ***c*** are modified according to the formula (Church and Gale, 1991):

$$c_g = (c+1)n_{c+1} / n_c \quad (4)$$

where c_g is a modified count replacing ***c*** in subsequent RF estimates, and n_c is the number of events having count ***c***.

Kneser-Ney (modified): an absolute discounting variant with

$$P_{KN}(\underline{s} \mid \underline{t}) = \frac{c(\underline{s},\underline{t}) - D}{c(\underline{t})} + \alpha(\underline{t}) \times P_b(\underline{s}) \quad (5)$$

where

$$\alpha(\underline{t}) = D \times n_{1+}(*,\underline{t}) / c(\underline{t}), \text{ and}$$
$$P_b(\underline{s}) = n_{1+}(\underline{s},*) / \sum_{\underline{s}} n_{1+}(\underline{s},*).$$

Here, $n_{1+}(*,\underline{t})$ is the number of unique source-language phrases **t** is aligned with; $n_{1+}(\underline{s},*)$ has an analogous definition. $P_{KN}(\mathbf{t}|\mathbf{s})$ is defined symmetrically. Kneser-Ney gives a bonus to phrase pairs (**s**,**t**) such that **s** and **t** have been aligned to many different phrases. Modified Kneser-Ney defines different discounts D depending on the value of c(**s**,**t**). We used "KN3", where D_1 is used when c(**s**,**t**) = 1, D_2 when c(**s**,**t**) = 2, and D_3 when c(**s**,**t**) ≥ 3. D_i values can be tuned or set by formula (we tried both without seeing a difference).

Low-Frequency Indicator (LF): (Mauser *et al.*, 2007) introduced these low-frequency features. Let

$$N[c(\underline{s},\underline{t}) \le \tau] = \begin{cases} 1 & if\ c(\underline{s},\underline{t}) \le \tau; \\ 0 & otherwise. \end{cases} \quad (6)$$

This feature "punishes" low-frequency phrase pairs. In the paper (Mauser *et al.*, 2007), three different low-frequency features were used, with the three values of τ lying in the interval between 0.9 and 3.0 (the system in the paper allows fractional values of c(**s**,**t**)).

3 Unpacking and Transforming Feature Functions

Taking just the two RF features from Equation (3), we have:

$$S_{RF} = \lambda_1 \times log[P_{RF}(\underline{t}|\underline{s})] + \lambda_2 \times log[P_{RF}(\underline{s}|\underline{t})]$$
$$= \lambda_1 \times (log[c(\underline{s},\underline{t})] - log[c(\underline{s})]) + \lambda_2 \times (log[c(\underline{s},\underline{t})] - log[c(\underline{t})])$$
$$= (\lambda_1 + \lambda_2) \times log[c(\underline{s},\underline{t})] - \lambda_1 \times log[c(\underline{s})] - \lambda_2 \times log[c(\underline{t})].$$

This is a combination of three terms, with an additive constraint. Wouldn't it be simpler to fit the following expression:

$$S_{RF} = \lambda_1 \times log[c(\underline{s},\underline{t})] + \lambda_2 \times log[c(\underline{s})] + \lambda_3 \times log[c(\underline{t})]? \quad (7)$$

MERT can still choose λ's that are equivalent to using the two RF estimates. In principle, nothing is lost and a degree of freedom, achieved by dropping the additive constraint, is gained (but this extra degree of freedom may lead to search errors).

The obvious objection is that this "3-count" replacement for the two RF features doesn't model probabilities. However, the inclusion of $P_{RF}(\underline{t}|\underline{s})$ among features can't be justified probabilistically either. Originally, the objective function for SMT was derived via Bayes's Theorem as $P(\underline{s}|\underline{t})\times P(\underline{t})$ (Brown *et al.*, 1993). The inclusion of $P(\underline{t}|\underline{s})$ happened later – it's a heuristic that defies Bayes's Theorem (Och and Ney, 2002).

Once the forward and backward estimates have been unpacked into their three constituent counts, these counts can be transformed and generalized slightly by adding or subtracting constants (while ensuring the logarithm is defined). Therefore, we have two different log-linear feature sets obtained from the three basic statistics:

3-count:

$$\{log[c(\underline{s},\underline{t})],\ log[c(\underline{s})],\ log[c(\underline{t})]\}$$

Generalized 3-count:

$$\{log[c(\underline{s},\underline{t})+k_1],\ log[c(\underline{s})+k_2],\ log[c(\underline{t})+k_3]\};$$

where $k_1, k_2, k_3 > -1$[1].

"Generalized 3-count" is related to two different forms of simple discounting, **absolute discounting** (AD) and **denominator discounting** (DD). Discounting is based on removing probability mass from observed events to account for events that might have been observed, but weren't. The simplest form of AD involves subtracting a fixed amount from co-occurrence counts in the RF formula: *e.g.*, $P_{AD}(\underline{s}|\underline{t}) = [c(\underline{s},\underline{t}) - D]/c(\underline{t})$, where $0<D<1$. AD has many variants and a long history of being discussed in the literature (Foster *et al.*, 2006; Chen and Goodman, 1998).

DD is another simple form of discounting. Here, one shrinks the RF fraction by adding a constant to the denominator: *e.g.*, $P_{DD}(\underline{s}|\underline{t}) = c(\underline{s},\underline{t})/[c(\underline{t})+E]$, where $E>0$. Strangely, we have not found any discussion of DD in the literature, though it is related to other known techniques. *E.g.*, suppose we have a prior $f(\underline{s}|\underline{t})$. To combine this with observations $c(\underline{s},\underline{t})$ and $c(\underline{t})$ using the maximum *a posteriori* (MAP) formula (Lee and Gauvain, 1993), we have

$$P_{MAP}(\underline{s}|\underline{t}) = [c(\underline{s},\underline{t}) + \tau\times f(\underline{s}|\underline{t})]/[c(\underline{t}) + \tau].$$

A reasonable prior estimate for $f(\underline{s}|\underline{t})$ might be $1/|\underline{s}|$, 1 over the number of possible source-language phrases. This is very, very small – nearly zero. Thus, we have

$$P_{MAP}(\underline{s}|\underline{t}) \approx c(\underline{s},\underline{t})/[c(\underline{t}) + \tau] \qquad (8)$$

– the DD estimate.

AD and DD have different effects on estimated probability distributions. Though both reduce estimated probabilities for "seen" events, AD has a "sharpening" effect, increasing the estimated ratio of the probability of the most likely seen events to the probability of the least likely ones. DD does not change the ratios between estimated probabilities for seen events.

To get pure AD with generalized 3-count, one sets $k_1 < 0$ and $k_2 = k_3 = 0$; to get pure DD, set $k_1 = 0$ and $k_2, k_3 > 0$. Thus, generalized 3-count is a generalization of both forms of discounting. In our experiments, to find k_1, k_2, and k_3, we first used MERT on the dev set to find the log-linear weights for all features, with k_1, k_2, and k_3 set to zero. Then we used downhill simplex (as in Zens and Ney, 2004) to estimate k_1, k_2, and k_3 with the log-linear weights fixed. Finally, we ran MERT again to obtain new log-linear weights.[2]

Moreover, we explore other transformations of phrase statistics: we developed an "unpacked and transformed" version of Kneser-Ney phrase table smoothing.

As will be seen in the experimental section, modified Kneser-Ney is an extremely effective technique. After we had done some initial experiments with it, we were thus highly motivated to "unpack" Kneser-Ney. However, the Kneser-Ney formula involves a sum of two terms; there is no easy way to unpack a log of a sum. We thus decided to try using the statistics that are unique to Kneser-Ney (*i.e.*, those statistics that aren't contained in 3-count) as separate features. We also tried generalizing them by adding constants, as we did for 3-count. We call $n_{1+}(*,\underline{t})$ the number of "events" for $\underline{t}$, and $n_{1+}(\underline{s},*)$ the number of "events" for $\underline{s}$. Thus we have:

[1] The minimal phrase pair joint or phrase marginal count is 1 in our system. Therefore, we set k's > -1 to ensure the logarithm is defined.

[2] We can't prove that it will converge after the second round of MERT. We did try to run one more round of downhill simplex and MERT in our experiments, but didn't get further improvements.

2-event (2EN):

$\{log[n_{1+}(*,\underline{t})], log[n_{1+}(\underline{s},*)]\}$

Generalized 2-event (Gen 2EN):

$\{log[n_{1+}(*,\underline{t})+k_1], log[n_{1+}(\underline{s},*)+k_2]\}$;
where $k_1, k_2 > -1$.

Downhill simplex and MERT are used to learn the k's as was described above for generalized 3-count. However, note that though in theory systems with 3-count or generalized 3-count features can learn $P_{RF}(\underline{t}|\underline{s})$ and $P_{RF}(\underline{s}|\underline{t})$, even systems with both the 3-count features and the 2-event features cannot learn the exact Kneser-Ney formula, because of the "log of sum" problem.

Finally, we devised a new feature that punishes low-frequency phrase pairs (as the Mauser *et al.* LF features described above do), but with a continuously decreasing penalty; it is a transformation of $c(\underline{s},\underline{t})$.

Enhanced Low-Frequency (ELF):

$$h_{ELF}(\underline{s},\underline{t}) = \log\left(e^{\frac{-1}{c(\underline{s},\underline{t})}}\right) = \frac{-1}{c(\underline{s},\underline{t})}. \quad (9)$$

At the log-linear level, ELF is equivalent to a penalty proportional to $1/c(\underline{s},\underline{t})$ being subtracted from the score. Phrase pairs with low joint counts are punished more than those phrase pairs with high joint counts. (We also tried on a dev set ELF-like features based on $1/c(\underline{s})$ and $1/c(\underline{t})$, but they did not yield consistently good results).

4 Experiments

4.1 System and lattice MERT details

We evaluated several new and several known techniques with our in-house phrase-based SMT system, whose decoder resembles Moses (Koehn *et al.*, 2007). In addition to phrase count features, all systems had forward and backward lexical probabilities, of the type described in (Zens and Ney, 2004), and lexicalized and distance-based distortion models. The LW estimates employed in our experiments are based on (Zens and Ney, 2004); Foster *et al.* (2006) found this to be the most effective lexical smoothing technique.

Weights on the feature functions are found by lattice MERT (LMERT) (Macherey *et al.*, 2008). These authors pruned the lattices output by their decoder; they also aggregated lattices over iterations (clarified via personal communication with W. Macherey). By contrast, an earlier version of LMERT employed by our group (Larkin *et al.* 2010) did not involve pruning or aggregation. Initially, we followed the Larkin *et al.* algorithm: this provides rapid convergence to reasonable optima. However, we decided that some aggregation should be tried to discourage random walk behaviour. In experiments for this paper, we found that without lattice aggregation, adding features led to worse optima on dev sets (despite the possibility of giving the new features zero or negligible weights).

Aggregation of lattices requires pruning because without it, successive iterations cause the memory requirements for the lattice to grow linearly. The search time will increase at a rate worse than quadratic, because the whole lattice must be scanned for each linemax operation. We implemented pruning ourselves: we kept only lattice arcs that were represented in a convex hull resulting from some linemax operation in a previous MERT iteration. The same answer as before would be produced if the MERT iteration were run with the pruned lattice - except that MERT would be much faster. Not all the convex hull is reproducible, but the part of it that was actually seen is reproducible. In practice, this form of pruning seems to address the addition-of-feature problem discussed above.

Thus, our revised LMERT searches the complete lattices from the latest decoder run combined with the pruned lattices representing the seen portions of the convex hull from all past iterations. Though the old convex hull may no longer be part of the current convex hull because of changes to the feature weights, it is a reminder of a part of the search space already seen. This solves the memory and speed problems and provides better performance.

4.2 Data

Results were obtained for Chinese-to-English (CE), and French-to-English (FE). There were two CE data conditions. The first is the *small data* condition where only the *FBIS*[3] corpus (10.5M target words) is used to train the translation model. For this condition, we built the phrase table using two phrase extractors: the in-house one extracts phrase pairs from merged counts of symmetrized IBM model 2 (Brown *et al.*, 1993) and HMM (Vogel *et al.*, 1996) word alignments, while the other one extracts phrase pairs from GIZA++ (Och and Ney 2003) IBM model 4 word alignments (in all other experiments, we only used the in-house extractor). The second is the *large data* condition where the pa-

[3] LDC2003E14

rallel training data are from the NIST[4] 2009 CE evaluation (112.6M target words). We used the same two language models (LMs) for both CE conditions: a 5-gram LM trained on the target side of the *large data* corpus, and a 6-gram LM trained on the English *Gigaword v4* corpus.

We used the same development and test sets for the two CE data conditions. The development set comprises mainly data from the NIST 2005 test set, and also some balanced-genre web-text from the NIST training material. Evaluation was performed on the NIST 2006 and 2008 test sets.

For FE, we used WMT 2010[5] FE data sets. Parallel *Europarl* data are used for training (47M English words); WMT Newstest 2008 is used as the dev set and WMT Newstest 2010 is the evaluation set. Two 5-gram LMs are used for FE: one is the English side of the parallel data, and the other the English side of *GigaFrEn*.

Our in-house system is a descendant of the (Foster *et al.*, 2006) system, but outperforms it by roughly 5 BLEU over a range of MT tasks. The main reasons for this are that the current version of the system carries out word alignment using both IBM 2 and HMM models instead of just IBM 2 ones, that it uses both lexicalized and distance-based distortion instead of just the latter, that it uses 5- and 6-grams instead of trigrams, and that lattice MERT is used for tuning weights instead of N-best MERT.

4.3 Results

Tables 1-3 show experimental results, arranged by the overall effectiveness of techniques. The baseline has the standard forward and backward relative frequencies (RF). We tried a combination of the two RF features with three low-frequency indicator features (RF+LF3), the two RF features with the ELF feature described earlier (RF+ELF), Good-Turing (GT), and the 3-count feature set described earlier (3CT) and its generalized form (Gen 3-CT); we also tried modified Kneser-Ney with three different discounts (KN3), and the same version of Kneser-Ney with ELF (KN3+ELF). For the small CE condition with the in-house extractor, we also tried 2-event (2EN) and generalized 2-event (Gen 2EN) (we unfortunately ran out of time to test these EN features in all conditions). In the tables, after the abbreviation for each system, we give in brackets the number of log-linear plus other weights that must be tuned for the non-lexical phrase count component of each: *e.g.*, KN3 has two probability estimates, with associated log-linear weights λ_1 and λ_2 tuned by MERT, and three discounts D_1, D_2, and D_3 (shared by forward and backward probabilities), giving (2+3) weights. Following (Koehn, 2004), we use the bootstrap-resampling test to do significance testing. In **Table 1-3**, Symbols ** or * indicates that the result is significant better than the baseline at level p<0.01 or p<0.05 respectively.

In-house extractor system (#wts)	**NIST test**		**Mean**	**Δ**
	2006	2008		
RF (2+0)	29.85	23.57	26.71	N/A
3CT (3+0)	29.48	23.24	26.36	-0.35
RF+LF3 (5+0)	29.87	23.60	26.74	+0.03
GT (2+0)	29.91	23.61	26.76	+0.05
RF+ELF (3+0)	30.46**	24.16**	27.31	+0.60
Gen3CT (3+3)	30.21	23.62	26.92	+0.21
Gen3CT+2EN (5+3)	30.47**	23.90*	27.19	+0.48
Gen3CT+Gen2EN (5+5)	30.52**	24.13**	27.33	+0.62
KN3 (2+3)	30.76**	24.36**	27.56	+0.85
KN3+ELF (3+3)	**30.91****	**24.53****	**27.72**	**+1.01**
GIZA++ extractor system (#wts)	**NIST test**		**Mean**	**Δ**
	2006	2008		
RF (2+0)	30.05	23.48	26.77	N/A
3CT (3+0)	30.68*	23.63	27.16	+0.39
RF+LF3 (5+0)	30.42	23.69	27.06	+0.29
GT (2+0)	30.99**	23.86*	27.43	+0.66
RF+ELF (3+0)	31.14**	23.92*	27.53	+0.76
Gen 3CT (3+3)	31.28**	24.05**	27.67	+0.90
KN3 (2+3)	31.67**	24.46**	28.07	+1.30
KN3+ELF (3+3)	**31.77****	**24.59****	**28.18**	**+1.41**

Table 1: CE small data (BLEU% scores). In brackets the number of log-linear plus other weights that must be tuned for the non-lexical phrase count component of each system. Symbols ** or * indicates that the result is significantly better than the baseline at level p<0.01 or p<0.05 respectively. In brackets are the number of log-linear plus the number of other weights that must be tuned for the non-lexical phrase count component of each system.

In-house extractor system (#wts)	**NIST test**		**Mean**	**Δ**
	2006	2008		
RF (2+0)	33.18	26.76	29.97	N/A
3CT (3+0)	33.31	26.98	30.15	+0.18
RF+LF3 (5+0)	33.56*	27.09*	30.33	+0.36
GT (2+0)	34.00**	27.29*	30.65	+0.68
RF+ELF (3+0)	33.87*	27.38*	30.63	+0.66
Gen 3CT (3+3)	34.04**	27.38*	30.71	+0.74

[4] http://www.nist.gov/speech/tests/mt (http://www.itl.nist.gov/iad/mig/tests/mt/2009/MT09_ConstrainedResources.pdf provides the list of resources from which *large data* was drawn).

[5] http://www.statmt.org/wmt10/

KN3 (2+3)	34.38**	27.80**	31.09	+1.12
KN3+ELF (3+3)	**34.49****	**27.99****	**31.24**	**+1.27**

Table 2: CE large data (BLEU scores)

3CT performs no worse than RF, and perhaps better (it has a higher score for three of the four systems). 3CT usually doesn't do quite as well as RF+LF3; GT does better than both. Generalized 3CT and RF+ELF are more or less tied, and both are clearly superior to RF, RF+LF3, 3CT, and GT. Best of all are the two Kneser-Ney variants, with KN3+ELF doing better than KN3. In the small CE condition with the in-house extractor, Gen 3-CT combined with two EN variants that both "unpack" Kneser-Ney, outperforms everything except the variants with exact Kneser-Ney; in this case, "unpacking" an estimator (Kneser-Ney) seems to be a bad idea. Techniques with more tuning weights tend to perform better (the best-performing one has 6 weights, while the RF baseline has 2), though there are exceptions.

In-house extractor system (#wts)	**WMT 2010 test**	**Δ**
RF (2+0)	26.32	N/A
3CT (3+0)	26.60*	+0.28
RF+LF3 (5+0)	26.66*	+0.34
GT (2+0)	26.69*	+0.37
RF+ELF (3+0)	26.87**	+0.55
Gen 3CT (3+3)	26.78*	+0.46
KN3 (2+3)	26.88**	+0.56
KN3+ELF (3+3)	**26.95****	**+0.63**

Table 3: FE WMT (BLEU scores)

4.4 Discussion

The results given here give some support to the idea of unpacking and transforming the statistics that make up probability estimators in phrase-based SMT systems. In the experiments described above, "generalized 3-count", derived from statistics hidden in forward and backward conditional probabilities, performs quite well. Presumably, this estimator performs better than the two conventional RF features because of the extra degrees of freedom. ELF – a transformation of c(**s**,**t**) – also performs very well. Presumably, ELF does better than the low-frequency (LF) features of Mauser *et al*. (2007) because it is continuous rather than discrete.

On the other hand, modified Kneser-Ney is still the most effective standalone technique, as it was in (Foster *et al*., 2006). If one looks at the number of tunable weights in brackets after the name of each technique in **Tables 1-3**, one sees that the success of modified Kneser-Ney can't be attributed to its having more degrees of freedom than competing techniques: *e.g*., although it only has 5 weights, it consistently beats generalized 3-count, which has 6 weights (and in the topmost set of results, it beats a technique that has 8 weights and another that has 10 weights).

Disappointingly, unpacking modified Kneser-Ney did not yield improved performance. This may have been because the way in which we unpacked it was not a true generalization (the system cannot reverse the unpacking by learning a certain combination of weights, as it can for "3-count" and "generalized 3-count"). The best combination in all conditions was Kneser-Ney with ELF (KN3+ELF).

Surprisingly, the impact of the techniques we tried does **not** decrease with the amount of training data. Over the two CE conditions (with in-house phrase extractor), the gain for all techniques over the RF baseline is bigger in the large data condition: *e.g.*, KN3+ELF yields a remarkable 1.27 BLEU improvement over the baseline for large data CE. Gains on FE, which has an intermediate amount of training data, are only moderate. We speculate that these techniques become more important as data quality gets worse (even if the amount of data increases): a high proportion of CE small and of WMT data, but a low proportion of CE large data, is of good quality.

5 Conclusion and Future Work

In this paper, we have shown that the 3-count features, an "unpacked" version of the RF features, obtained results comparable to or perhaps better than those of the RF ones. Moreover, the generalized 3-count features achieved further improvement. We have also shown that a simple feature we call ELF – a transformation of c(**s**,**t**) – performs well. However, the unpacked and transformed version of Kneser-Ney (KN) smoothing did not perform as well as the original KN smoothing.

The comparison of phrase table smoothing techniques given here should be of general interest. At a minimum, all SMT practitioners should be aware that by implementing a simple technique like modified Kneser-Ney smoothing of phrase tables, they may obtain BLEU gains in the range +0.6-1.3.

In future work, we plan to try "unpacking" Good-Turing counts of counts, lexical weights,

and lexicalized distortion. The results above show that this kind of experimentation can have a good payoff, and it is now practical to train SMT systems with very many features (Chiang *et al*., 2009). We will also study the interaction between families of techniques: those for discounting/smoothing as in this paper, those for growing more focused phrase tables (Wuebker *et al*., 2010) and those for pruning phrase tables (Johnson *et al*., 2007). Though these three approaches partially overlap (all "punish" low-frequency phrase pairs), a combination may be more powerful than any of them alone.

References

P. Brown, S. Della Pietra, V. Della J. Pietra, and R. Mercer. "The mathematics of Machine Translation: Parameter estimation". *Computational Linguistics*, 19(2):263-312, June 1993.

S. Chen and J. Goodman. "An empirical study of smoothing techniques for language modeling". *Technical Report TR-10-98*, Computer Science Group, Harvard University, 1998.

D. Chiang, K. Knight, and W. Wang. "11,001 New Features for Statistical Machine Translation". *Proc. ACL*, pp 218–226, Boulder, Colorado, 2009.

K. Church and W. Gale. "A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams". *Computer speech and language*, 5(1):19–54, 1991.

G. Foster, R. Kuhn, and H. Johnson. "Phrasetable smoothing for statistical machine translation". *Proc. EMNLP*, pp. 53-61, Sydney, Australia, July 2006.

H. Johnson, J. Martin, G. Foster, and R. Kuhn. "Improving Translation Quality by Discarding Most of the Phrasetable". *Proc. EMNLP*, Prague, Czech Republic, June 28-30, 2007.

P. Koehn, F. J. Och, D. Marcu. "Statistical Phrase-Based Translation". *Proc. HLT-NAACL*, 2003.

P. Koehn. "Statistical significance tests for machine translation evaluation." *Proc. of EMNLP*, pp. 388–395. Barcelona, Spain. July, 2004.

P. Koehn, H. Hoang, *et al*. "Moses: Open Source Toolkit for Statistical Machine Translation". *Proc. ACL,* pp. 177-180, Prague, Czech Republic, June 2007.

S. Larkin, B. Chen, *et al.* "Lessons from NRC's Portage System at WMT 2010". *ACL Workshop on SMT and Metrics MATR*, pp. 133-138, Uppsala, Sweden, July 2010.

C.-H. Lee and J.-L. Gauvain. "Speaker adaptation based on MAP estimation of HMM parameters". *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, V. 2, pp. 558-561, 1993.

W. Macherey, F. Och, I. Thayer, and J. Uszkoreit. "Lattice-based Minimum Error Rate Training for Statistical Machine Translation". *Proc. EMNLP*, pp. 725-734, Honolulu, Hawaii, October 2008.

A. Mauser, D. Vilar, G. Leusch, Y. Zhang and H. Ney. "The RWTH Machine Translation System for IWSLT 2007". *Int. Workshop on Spoken Language Translation (IWSLT)*, pp. 161-168, Trento, Italy, October 2007.

F. Och. "Minimum error rate training for statistical machine translation". *Proc. ACL*, Sapporo, Japan, July 2003.

F. Och and H. Ney. "Discriminative training and maximum entropy models for statistical machine translation". *Proc. ACL*, Philadelphia, July 2002.

F. J. Och and H. Ney. "A systematic comparison of various statistical alignment models". *Computational Linguistics*, 29(1):19–51, 2003.

S. Vogel, H. Ney, and C. Tillmann. "HMM based word alignment in statistical translation". *Proc. COLING*, 1996.

J. Wuebker, A. Mauser, and H. Ney. "Training Phrase Translation Models with Leaving-One-Out". *Proc. ACL*, pp. 475-484, Uppsala, Sweden, July 2010.

R. Zens and H. Ney. "Improvements in phrase-based statistical machine translation". *Proc. NAACL/HLT*, pp. 257-264, Boston, USA, May 2004.