Capitalizing Machine Translation

Wei Wang and **Kevin Knight** and **Daniel Marcu**
Language Weaver, Inc.
4640 Admiralty Way, Suite 1210
Marina del Rey, CA, 90292
{wwang, kknight, dmarcu}@languageweaver.com

Abstract

We present a probabilistic bilingual capitalization model for capitalizing machine translation outputs using conditional random fields. Experiments carried out on three language pairs and a variety of experiment conditions show that our model significantly outperforms a strong monolingual capitalization model baseline, especially when working with small datasets and/or European language pairs.

1 Introduction

Capitalization is the process of recovering case information for texts in lowercase. It is also called truecasing (Lita et al., 2003). Usually, capitalization itself tries to improve the legibility of texts. It, however, can affect the word choice or order when interacting with other models. In natural language processing, a good capitalization model has been shown useful for tasks like name entity recognition, automatic content extraction, speech recognition, modern word processors, and machine translation (MT).

Capitalization can be viewed as a sequence labeling process. The input to this process is a sentence in lowercase. For each lowercased word in the input sentence, we have several available capitalization tags: initial capital (IU), all uppercase (AU), all lowercase (AL), mixed case (MX), and all having no case (AN). The output of capitalization is a capitalization tag sequence. Associating a tag in the output with the corresponding lowercased word in the input results in a surface form of the word. For example, we can tag the input sentence "click ok to save your changes to /home/doc." into "click_IU ok_AU to_AL save_AL your_AL changes_AL to_AL /home/doc_MX ._AN", getting the surface form "Click OK to save your changes to /home/DOC .".

A capitalizer is a tagger that recovers the capitalization tag for each input lowercased word, outputting a well-capitalized sentence. Since each lowercased word can have more than one tag, and associating a tag with a lowercased word can result in more than one surface form (e.g., /home/doc_MX can be either /home/DOC or /home/Doc), we need a capitalization model to solve the capitalization ambiguities. For example, Lita et al. (2003) use a trigram language model estimated from a corpus with case information; Chelba and Acero (2004) use a maximum entropy Markov model (MEMM) combining features involving words and their cases.

Capitalization models presented in most previous approaches are monolingual because the models are estimated only from monolingual texts. However, for capitalizing machine translation outputs, using only monolingual capitalization models is not enough. For example, if the sentence "click ok to save your changes to /home/doc ." in the above example is the translation of the French sentence "CLIQUEZ SUR OK POUR ENREGISTRER VOS MODIFICATIONS DANS /HOME/DOC .", the correct capitalization result should probably be "CLICK OK TO SAVE YOUR CHANGES TO /HOME/DOC .", where all words are in all upper-case. Without looking into the case

Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, pages 1–8,
New York, June 2006.

of the MT input, we can hardly get the correct capitalization result.

Although monolingual capitalization models in previous work can apply to MT output, a bilingual model is more desirable. This is because MT outputs usually strongly preserve case from the input, and because monolingual capitalization models do not always perform as well on badly translated text as on well-formed syntactic texts.

In this paper, we present a bilingual capitalization model for capitalizing machine translation outputs using conditional random fields (CRFs) (Lafferty et al., 2001). This model exploits case information from both the input sentence (source) and the output sentence (target) of the MT system. We define a series of feature functions to incorporate capitalization knowledge into the model.

Experimental results are shown in terms of BLEU scores of a phrase-based SMT system with the capitalization model incorporated, and in terms of capitalization precision. Experiments are performed on both French and English targeted MT systems with large-scale training data. Our experimental results show that the CRF-based bilingual capitalization model performs better than a strong baseline capitalizer that uses a trigram language model.

2 Related Work

A simple capitalizer is the 1-gram tagger: the case of a word is always the most frequent one observed in training data, with the exception that the sentence-initial word is always capitalized. A 1-gram capitalizer is usually used as a baseline for capitalization experiments (Lita et al., 2003; Kim and Woodland, 2004; Chelba and Acero, 2004).

Lita et al. (2003) view capitalization as a lexical ambiguity resolution problem, where the lexical choices for each lowercased word happen to be its different surface forms. For a lowercased sentence e, a trigram language model is used to find the best capitalization tag sequence T that maximizes $p(T, e) = p(E)$, resulting in a case-sensitive sentence E. Besides local trigrams, sentence-level contexts like sentence-initial position are employed as well.

Chelba and Acero (2004) frame capitalization as a sequence labeling problem, where, for each lowercased sentence e, they find the label sequence T that maximizes $p(T|e)$. They use a maximum entropy Markov model (MEMM) to combine features of words, cases and context (i.e., tag transitions).

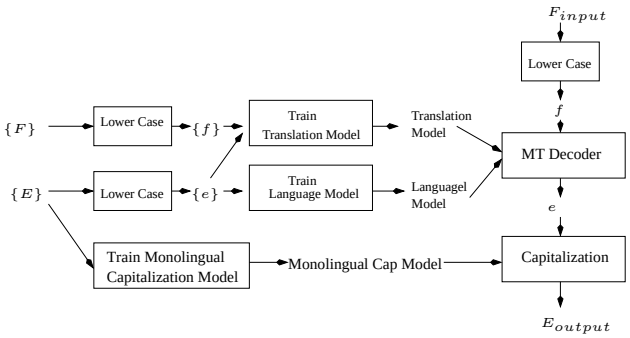

Figure 1: The monolingual capitalization scheme employed by most statistical MT systems.

Gale et al. (1994) report good results on capitalizing 100 words. Mikheev (1999) performs capitalization using simple positional heuristics.

3 Monolingual Capitalization Scheme

Translation and capitalization are usually performed in two successive steps because removing case information from the training of translation models substantially reduces both the source and target vocabulary sizes. Smaller vocabularies lead to a smaller translation model with fewer parameters to learn. For example, if we do not remove the case information, we will have to deal with at least nine probabilities for the English-French word pair (click, cliquez). This is because either "click" or "cliquez" can have at least three tags (IU, AL, AU), and thus three surface forms. A smaller translation model requires less training data, and can be estimated more accurately than otherwise from the same amount of training data. A smaller translation model also means less memory usage.

Most statistical MT systems employ the monolingual capitalization scheme as shown in Figure 1. In this scheme, the translation model and the target language model are trained from the lowercased corpora. The capitalization model is trained from the case-sensitive target corpus. In decoding, we first turn input into lowercase, then use the decoder to generate the lowercased translation, and finally ap-

HYDRAULIC HEADER TILT CYLINDER KIT Kit de vérin d'inclinaison hydraulique de la plate-forme
haut-parleur avant droit + HAUT-PARLEUR AVANT DROIT +
Seat Controls, Standard COMMANDES DU SIGE, STANDARD
loading a saved legend Chargement d'une légende sauvegarde

Table 1: Errors made by monolingual capitalization model. Each row contains a pair of MT input and MT output.

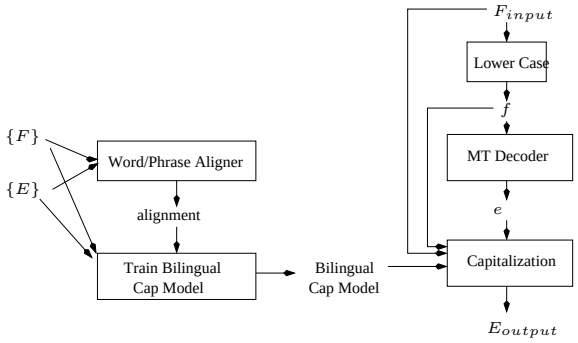

Figure 2: A bilingual capitalization scheme.

ply the capitalization model to recover the case of the decoding output.

The monolingual capitalization scheme makes many errors as shown in Table 1. Each cell in the table contains the MT-input and the MT-output. These errors are due to the capitalizer does not have access to the source sentence.

Regardless, estimating mixed-cased translation models, however, is a very interesting topic and worth future study.

4 Bilingual Capitalization Model

4.1 The Model

Our probabilistic bilingual capitalization model exploits case information from both the input sentence to the MT system and the output sentence from the system (see Figure 2). An MT system translates a *capitalized* sentence F into a lowercased sentence e. A statistical MT system can also provide the alignment A between the input F and the output e; for example, a statistical phrase-based MT system could provide the phrase boundaries in F and e, and also the alignment between the phrases.[1]

[1]We shall explain our capitalization model within the phrase-based SMT framework, the model, however, could be adapted to syntax-based machine translation, too. To this end, the translational correspondence is described within a translation rule, i.e., (Galley et al., 2004) (or a synchronous production), rather than a translational phrase pair; and the training data will be derivation forests, instead of the phrase-aligned bilingual corpus.

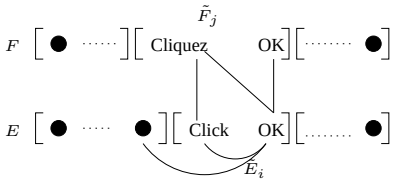

Figure 3: Alignment graph. Brackets mean phrase boundaries.

The bilingual capitalization algorithm recovers the capitalized sentence E from e, according to the input sentence F, and the alignment A. Formally, we look for the best capitalized sentence E^* such that

$$E^* = \arg\max_{E \in \mathbf{GEN}(e)} p(E|F, A) \quad (1)$$

where $\mathbf{GEN}(e)$ is a function returning the set of possible capitalized sentences consistent with e. Notice that e does not appear in $p(E|F, A)$ because we can uniquely obtain e from E. $p(E|F, A)$ is the capitalization model of concern in this paper.[2]

To further decompose the capitalization model $p(E|F, A)$, we make some assumptions. As shown in Figure 3, input sentence F, capitalized output E, and their alignment can be viewed as a graph. Vertices of the graph correspond to words in F and E. An edge connecting a word in F and a word in E corresponds to a word alignment. An edge between two words in E represents the dependency between them captured by monolingual n-gram language models. We also assume that both E and F have phrase boundaries available (denoted by the square brackets), and that A is the phrase alignment. In Figure 3, $\tilde{F}_j$ is the j-th phrase of F, $\tilde{E}_i$ is the i-th phrase of E, and they align to each other. We do not require a word alignment; instead we find it reasonable to think that a word in $\tilde{E}_i$ can be aligned to any

[2]The capitalization model $p(E|F, A)$ itself does not require the existence of e. This means that in principle this model can also be viewed as a capitalized translation model that performs translation and capitalization in an integrated step. In our paper, however, we consider the case where the machine translation output e is given, which is reflected by the the fact that **GEN**(e) takes e as input in Formula 1.

word in $\tilde{F}_j$. A probabilistic model defined on this graph is a Conditional Random Field. Therefore, it is natural to formulate the bilingual capitalization model using CRFs:[3]

$$p_{\overline{\lambda}}(E|F,A) = \frac{1}{Z(F,A,\overline{\lambda})} \exp\left(\sum_{i=1}^{I} \lambda_i f_i(E,F,A)\right) \quad (2)$$

where

$$Z(F,A,\overline{\lambda}) = \sum_{E \in \mathbf{GEN}(e)} \exp\left(\sum_{i=1}^{I} \lambda_i f_i(E,F,A)\right) \quad (3)$$

$f_i(E,F,A), i = 1...I$ are the I features, and $\overline{\lambda} = (\lambda_1, ..., \lambda_I)$ is the feature weight vector. Based on this capitalization model, the decoder in the capitalizer looks for the best E^* such that

$$E^* = \arg\max_{E \in \mathbf{GEN}(e,F)} \sum_{i=1}^{I} \lambda_i f_i(E,F,A) \quad (4)$$

4.2 Parameter Estimation

Following Roark et al. (2004), Lafferty et al. (2001) and Chen and Rosenfeld (1999), we are looking for the set of feature weights $\overline{\lambda}$ maximizing the regularized log-likelihood $LL_R(\overline{\lambda})$ of the training data $\{E^{(n)}, F^{(n)}, A^{(n)}, n = 1, ..., N\}$.

$$LL_R(\overline{\lambda}) = \sum_{n=1}^{N} \log p\left(E^{(n)}|F^{(n)}, A^{(n)}\right) - \frac{||\overline{\lambda}||^2}{2\sigma^2} \quad (5)$$

The second term at the right-hand side of Formula 5 is a zero-mean Gaussian prior on the parameters. σ is the variance of the Gaussian prior dictating the cost of feature weights moving away from the mean — a smaller value of σ keeps feature weights closer to the mean. σ can be determined by linear search on development data.[4] The use of the Gaussian prior term in the objective function has been found effective in avoiding overfitting, leading to consistently better results. The choice of LL_R as an objective function can be justified as maximum a-posteriori (MAP) training within a Bayesian approach (Roark et al., 2004).

[3]We chose CRFs over other sequence labeling models (i.e. MEMM) because CRFs have no label bias and we do not need to compute the partition function during decoding.

[4]In our experiment, we use an empirical value $\sigma = 0.5$ as in (Roark et al., 2004).

4.3 Feature Functions

We define features based on the alignment graph in Figure 3. Each feature function is defined on a word.

Monolingual language model feature. The monolingual LM feature of word E_i is the logarithm of the probability of the n-gram ending at E_i:

$$f_{\mathrm{LM}}(E_i, F, A) = \log p(E_i|E_{i-1}, ..., E_{i-n+1}) \quad (6)$$

p should be appropriately smoothed such that it never returns zero.

Capitalized translation model feature. Suppose E phrase "Click OK" is aligned to F phrase "Cliquez OK". The capitalized translation model feature of "Click" is computed as $\log p(\mathrm{Click}|\mathrm{Cliquez}) + \log p(\mathrm{Click}|\mathrm{OK})$. "Click" is assumed to be aligned to any word in the F phrase. The larger the probability that "Click" is translated from an F word, i.e., "Cliquez", the more chances that "Click" preserves the case of "Cliquez". Formally, for word E_i, and an aligned phrase pair $\tilde{E}_l$ and $\tilde{F}_m$, where $E_i \in \tilde{E}_l$, the capitalized translation model feature of E_i is

$$f_{\mathrm{cap\cdot t1}}(E_i, F, A) = \log \sum_{k=1}^{|\tilde{F}_m|} p(E_i|\tilde{F}_{m,k}) \quad (7)$$

$p(E_i|\tilde{F}_{m,k})$ is the capitalized translation table. It needs smoothing to avoid returning zero, and is estimated from a word-aligned bilingual corpus.

Capitalization tag translation feature. The feature value of E word "Click" aligning to F phrase "Cliquez OK" is $\log p(\mathrm{IU}|\mathrm{IU})p(\mathrm{click}|\mathrm{cliquez}) + \log p(\mathrm{IU}|\mathrm{AU})p(\mathrm{click}|\mathrm{ok})$. We see that this feature is less specific than the capitalized translation model feature. It is computed in terms of the tag translation probability and the lowercased word translation probability. The lowercased word translation probability, i.e., $p(\mathrm{click}|\mathrm{ok})$, is used to decide how much of the tag translation probability, i.e., $p(\mathrm{IU}|\mathrm{AU})$, will contribute to the final decision. The smaller the word translation probability, i.e., $p(\mathrm{click}|\mathrm{ok})$, is, the smaller the chance that the surface form of "click"

preserves case from that of "ok". Formally, this feature is defined as

$$f_{\text{cap·tag·t1}}(E_i, F, A) = \log \sum_{k=1}^{|\tilde{f}_m|} p(e_i|\tilde{f}_{m,k}) \times p(\tau(E_i)|\tau(\tilde{F}_{m,k})) \quad (8)$$

$p(e_i|\tilde{f}_{m,k})$ is the t-table over lowercased word pairs, which is the usual "t-table" in a SMT system. $p(\tau(E_i)|\tau(\tilde{F}_{m,k}))$ is the probability of a target capitalization tag given a source capitalization tag and can be easily estimated from a word-aligned bilingual corpus. This feature attempts to help when $f_{\text{cap}-\text{t1}}$ fails (i.e., the capitalized word pair is unseen). Smoothing is also applied to both $p(e_i|\tilde{f}_{m,k})$ and $p(\tau(E_i)|\tau(\tilde{F}_{m,k}))$ to handle unseen words (or word pairs).

Upper-case translation feature. Word E_i is in all upper case if all words in the corresponding F phrase $\tilde{F}_m$ are in upper case. Although this feature can also be captured by the capitalization tag translation feature in the case where an AU tag in the input sentence is most probably preserved in the output sentence, we still define it to emphasize its effect. This feature aims, for example, to translate "ABC XYZ" into "UUU VVV" even if all words are unseen.

Initial capitalization feature. An E word is initially capitalized if it is the first word *that contains* letters in the E sentence. For example, for sentence "• Please click the button" that starts with a bullet, the initial capitalization feature value of word "please" is 1 because "•" does not contain a letter.

Punctuation feature template. An E word is initially capitalized if it follows a punctuation mark. Non-sentence-ending punctuation marks like commas will usually get negative weights.

As one can see, our features are "coarse-grained" (e.g., the language model feature). In contrast, Kim and Woodland (2004) and Roark et al. (2004) use "fine-grained" features. They treat each n-gram as a feature for, respectively, monolingual capitalization and language modeling. Feature weights tuned at a fine granularity may lead to better accuracy, but they require much more training data, and result in much slower training speed, especially for large-scale learning problems. Coarse-grained features enable us to efficiently get the feature values from a very large training corpus, and quickly tune the weights on small development sets. For example, we can train a bilingual capitalization model on a 70 million-word corpus in several hours with the coarse-grained features presented above, but in several days with fine-grained n-gram count features.

4.4 The GEN Function

Function **GEN** generates the set of case-sensitive candidates from a lowercased token. For example $\mathbf{GEN}(\texttt{mt}) = \{\texttt{mt}, \texttt{mT}, \texttt{Mt}, \texttt{MT}\}$. The following heuristics can be used to reduce the range of **GEN**. The returned set of **GEN** on a lower-cased token w is the union of: (i) $\{w, AU(w), IU(w)\}$, (ii) $\{v|v \; is \; seen \; in \; training \; data \; and \; AL(v) = w\}$, and (iii) $\{\tilde{F}_{m,k}|AL(\tilde{F}_{m,k}) = AL(w)\}$. The heuristic (iii) is designed to provide more candidates for w when it is translated from a very strange input word $\tilde{F}_{m,k}$ in the F phrase $\tilde{F}_m$ that is aligned to the phrase that w is in. This heuristic creates good capitalization candidates for the translation of URLs, file names, and file paths.

5 Generating Phrase-Aligned Training Data

Training the bilingual capitalization model requires a bilingual corpus with phrase alignments, which are usually produced from a phrase aligner. In practice, the task of phrase alignment can be quite computationally expensive as it requires to translate the entire training corpus; also a phrase aligner is not always available. We therefore generate the training data using a naïve phrase aligner (NPA) instead of resorting to a real one.

The input to the NPA is a word-aligned bilingual corpus. The NPA stochastically chooses for each sentence pair one segmentation and phrase alignment that is consistent with the word alignment. An aligned phrase pair is consistent with the word alignment if neither phrase contains any word aligning to a word outside the other phrase (Och and Ney, 2004). The NPA chunks the source sentence into phrases according to a probabilistic distribution over source phrase lengths. This distribution can be obtained from the trace output of a phrase-based MT

Languages	Entire Corpus (#W) Training	Dev	Test-Prec.	Test-BLEU (#sents)
E→F (IT)	62M	13K	15K	763
F→E (news)	144M	11K	22K	241
C→E (news)	50M	8K	17K	919

Table 2: Corpora used in experiments.

decoder on a small development set. The NPA has to retry if the current source phrase cannot find any consistent target phrase. Unaligned target words are attached to the left phrase. Heuristics are employed to prevent the NPA from not coming to a solution. Obviously, the NPA is a special case of the phrase extractor in (Och and Ney, 2004) in that it considers only one phrase alignment rather than all possible ones.

Unlike a real phrase aligner, the NPA need not wait for the training of the translation model to finish, making it possible for parallelization of translation model training and capitalization model training. However, we believe that a real phrase aligner may make phrase alignment quality higher.

6 Experiments

6.1 Settings

We conducted capitalization experiments on three language pairs: English-to-French (E→F) with a bilingual corpus from the Information Technology (IT) domain; French-to-English (F→E) with a bilingual corpus from the general news domain; and Chinese-to-English (C→E) with a bilingual corpus from the general news domain as well. Each language pair comes with a training corpus, a development corpus and two test sets (see Table 2). Test-Precision is used to test the capitalization precision of the capitalizer on well-formed sentences drawn from genres similar to those used for training. Test-BLEU is used to assess the impact of our capitalizer on end-to-end translation performance; in this case, the capitalizer may operate on ungrammatical sentences. We chose to work with these three language pairs because we wanted to test our capitalization model on both English and French target MT systems and in cases where the source language has no case information (such as in Chinese).

We estimated the feature functions, such as the log probabilities in the language model, from the training set. Kneser-Ney smoothing (Kneser and Ney, 1995) was applied to features f_{LM}, $f_{\mathrm{cap{\cdot}t1}}$, and $f_{\mathrm{cap{\cdot}tag{\cdot}t1}}$. We trained the feature weights of the CRF-based bilingual capitalization model using the development set. Since estimation of the feature weights requires the phrase alignment information, we efficiently applied the NPA on the development set.

We employed two LM-based capitalizers as baselines for performance comparison: a unigram-based capitalizer and a strong trigram-based one. The unigram-based capitalizer is the usual baseline for capitalization experiments in previous work. The trigram-based baseline is similar to the one in (Lita et al., 2003) except that we used Kneser-Ney smoothing instead of a mixture.

A phrase-based SMT system (Marcu and Wong, 2002) was trained on the bitext. The capitalizer was incorporated into the MT system as a post-processing module — it capitalizes the lowercased MT output. The phrase boundaries and alignments needed by the capitalizer were automatically inferred as part of the decoding process.

6.2 BLEU and Precision

We measured the impact of our capitalization model in the context of an end-to-end MT system using BLEU (Papineni et al., 2001). In this context, the capitalizer operates on potentially ill-formed, MT-produced outputs.

To this end, we first integrated our bilingual capitalizer into the phrase-based SMT system as a post-processing module. The decoder of the MT system was modified to provide the capitalizer with the case-preserved source sentence, the lowercased translation, and the phrase boundaries and their alignments. Based on this information, our bilingual capitalizer recovers the case information of the lowercased translation, outputting a capitalized target sentence. The case-restored machine translations were evaluated against the target test-BLEU set. For comparison, BLEU scores were also computed for an MT system that used the two LM-based baselines.

We also assessed the performance of our capitalizer on the task of recovering case information for well-formed grammatical texts. To this end, we used the precision metric that counted the number of cor-

rectly capitalized words produced by our capitalizer on well-formed, lowercased input

$$precision = \frac{\#correctly\ capitalized\ words}{\#total\ words} \quad (9)$$

To obtain the capitalization precision, we implemented the capitalizer as a standalone program. The inputs to the capitalizer were triples of a case-preserved source sentence, a lowercased target sentence, and phrase alignments between them. The output was the case-restored version of the target sentence. In this evaluation scenario, the capitalizer output and the reference differ only in case information — word choices and word orders between them are the same. Testing was conducted on Test-Precision. We applied the NPA to the Test-Precision set to obtain the phrases and their alignments because they were needed to trigger the features in testing. We used a Test-Precision set that was different from the Test-BLEU set because word alignments were by-products only of training of translation models on the MT training data and we could not put the Test-BLEU set into the MT training data. Rather than implementing a standalone word aligner, we randomly divided the MT training data into three non-overlapping sets: Test-Precision set, CRF capitalizer training set and dev set.

6.3 Results

The performance comparisons between our CRF-based capitalizer and the two LM-based baselines are shown in Table 3 and Table 4. Table 3 shows the BLEU scores, and Table 4 shows the precision. The BLEU upper bounds indicate the ceilings that a perfect capitalizer can reach, and are computed by ignoring the case information in both the capitalizer outputs and the reference. Obviously, the precision upper bounds for all language pairs are 100%.

The precision and end-to-end BLEU based comparisons show that, for European language pairs, the CRF-based bilingual capitalization model outperforms significantly the strong LM-based baseline. We got more than one BLEU point improvement on the MT translation between English and French, a 34% relative reduction in capitalization error rate for the French-to-English language pair, and a 42% relative error rate reduction for the English-to-French language pair. These results show that source language information provides significant help for capitalizing machine translation outputs. The results also show that when the source language does not have case, as in Chinese, the bilingual model equals a monolingual one.

The BLEU difference between the CRF-based capitalizer and the trigram one were larger than the precision difference. This indicates that the CRF-based capitalizer performs much better on non-grammatical texts that are generated from an MT system due to the bilingual feature of the CRF capitalizer.

6.4 Effect of Training Corpus Size

The experiments above were carried out on large data sets. We also conducted experiments to examine the effect of the training corpus size on capitalization precision. Figure 4 shows the effects. The experiment was performed on the E→F corpus. The bilingual capitalizer performed significantly better when the training corpus size was small (e.g., under 8 million words). This is common in many domains: when the training corpus size increases, the difference between the two capitalizers decreases.

7 Conclusions

In this paper, we have studied how to exploit bilingual information to improve capitalization performance on machine translation output, and evaluated the improvement over traditional methods that use only monolingual language models.

We first presented a probabilistic bilingual capitalization model for capitalizing machine translation outputs using conditional random fields. This model exploits bilingual capitalization knowledge as well as monolingual information. We defined a series of feature functions to incorporate capitalization knowledge into the model.

We then evaluated our CRF-based bilingual capitalization model both on well-formed texts in terms of capitalization precision, and on possibly ungrammatical end-to-end machine translation outputs in terms of BLEU scores. Experiments were performed on both French and English target MT systems with large-scale training data. Our experimental results showed that the CRF-based bilingual cap-

Translation	BLEU Scores			
	Unigram Capitalizer	Trigram Capitalizer	CRF-based Capitalizer	Upper Bound
F→E	24.96	26.73	**27.92**	28.85
E→F	32.63	34.66	**36.10**	36.17
C→E	23.81	25.92	**25.89**	-

Table 3: Impact of CRF-based capitalizer on end-to-end translation performance compared with two LM-based baselines.

Translation	Capitalization Precision (%)		
	Unigram capitalizer	Trigram capitalizer	CRF-based capitalizer
F→E	94.03	98.79	**99.20**
E→F	91.52	98.47	**99.11**
C→E	90.77	96.40	**96.76**

Table 4: Impact of CRF-based capitalizer on capitalization precision compared with two LM-based baselines.

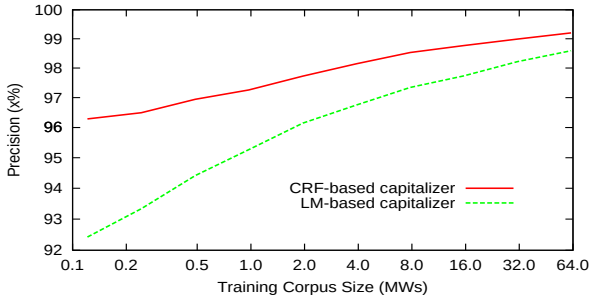

Figure 4: Capitalization precision with respect to size of training corpus. LM-based capitalizer refers to the trigram-based one. Results were on E→F corpus.

italization model performs significantly better than a strong baseline, monolingual capitalizer that uses a trigram language model.

In all experiments carried out at Language Weaver with customer (or domain specific) data, MT systems trained on lowercased data coupled with the CRF bilingual capitalizer described in this paper consistently outperformed both MT systems trained on lowercased data coupled with a strong monolingual capitalizer and MT systems trained on mixed-cased data.

References

Ciprian Chelba and Alex Acero. 2004. Adaptation of maximum entroy capitalizer: Little data can help a lot. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Barcelona, Spain.

Stanley Chen and Ronald Rosenfeld. 1999. A Gaussian prior for smoothing Maximum Entropy models. Technical Report CMUCS-99-108, Carnegie Mellon University.

William A. Gale, Kenneth W. Church, and David Yarowsky. 1994. Discrimination decisions for 100,000-dimensional spaces. In *Current issues in computational linguistics*.

M. Galley, M. Hopkins, K. Knight, and D. Marcu. 2004. What's in a Translation Rule? In *Proceedings of the Human Language Technology Conference and the North American Association for Computational Linguistics (HLT-NAACL)*, Boston, Massachusetts.

Ji-Hwan Kim and Philip C. Woodland. 2004. Automatic capitalization generation for speech input. *Computer Speech and Language*, 18(1):67–90, January.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 1995*, pages 181–184, Detroit, Michigan. IEEE.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fi elds: Probabilistic models for segmentation and labeling sequence data.

Lucian Vlad Lita, Abe Ittycheriah, Salim Roukos, and Nanda Kambhatla. 2003. tRuEcasIng. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Sapporo, Japan, July.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Philadelphia, PA.

A. Mikheev. 1999. A knowledge-free method fro capitalized word disambiguation. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL)*, College Park, Maryland, June.

Franz Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4).

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: A method for automatic evaluation of Machine Translation. Technical Report RC22176, IBM, September.

Brian Roark, Murat Saraclar, Michael Collins, and Mark Johnson. 2004. Discriminative language modeling with conditional random fi eld and the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, Barcelona, Spain.