Machine Translation for Human Translators

Michael Denkowski
Language Technologies Institute
School of Computer Science
Carnegie Mellon University

Ph.D. Thesis Proposal
May 30, 2013

Thesis Committee:
Alon Lavie (chair), Carnegie Mellon University
Chris Dyer, Carnegie Mellon University
Jaime Carbonell, Carnegie Mellon University
Gregory Shreve, Kent State University

Abstract

While machine translation is sometimes sufficient for conveying information across language barriers, many scenarios still require precise human-quality translation that MT is currently unable to deliver. Governments and international organizations such as the United Nations require accurate translations of content dealing with complex geopolitical issues. Community-driven projects such as Wikipedia rely on volunteer translators to bring accurate information to diverse language communities. As the amount of data requiring translation has continued to increase, the idea of using machine translation to improve the speed of human translation has gained interest. In the frequently employed practice of post-editing, a machine translation system outputs an initial translation and a human translator edits it for correctness, ideally saving time over translating from scratch. While general improvements in MT quality have led to productivity gains with this technique, there has been little work on designing translation systems specifically for post-editing.

In this work, we propose improvements to key components of statistical machine translation systems aimed at directly reducing the amount of work required from human translators. We propose casting MT for post-editing as an online learning task where new training instances are created as humans edit system output, introducing an online translation model that immediately learns from post-editor feedback. We propose an extended translation feature set that allows this model to learn from multiple translation contexts over time as data sources become more reliable. We propose an automatic evaluation metric that scores hypothesis-reference pairs according to several statistics that are directly interpretable as measuring of post-editing effort. Our metric can be used to optimize translation systems in scenarios where standard metrics break down, select optimal system configurations for post-editing, and provide insight into the properties of translation quality that are most important for minimizing editing effort. Our online translation models and evaluation metrics are compatible with standard decoders and optimization algorithms.

To evaluate the impact of our post-editing-targeted translation system, we propose a series of experiments that use a web-based framework to collect several types of highly accurate data from human translators. We discuss MT for post-editing as a distinct task and present the results of initial post-editing experiments. We finally outline an experimental setup for collecting valuable data that will be used to evaluate the impact of our online translation models and optimization metrics on human editing requirements.

Contents

Chapter 1

Background

1.1 The Mechanics of Phrase-Based Machine Translation

Faced with the task of translating a French sentence into English, a human translator has the ability to read the original sentence, call on knowledge of source language syntax and semantics to discern the meaning, and write out a grammatically correct sentence in the target language that conveys the same meaning. Years of reading and writing both languages allow the translator to be sensitive to nuances such as idioms, tone, and context when crafting a polished, meaning equivalent translation. Given the same task, a machine translation system with no inherent knowledge of human language cannot replicate this process. However, when provided with large amounts of bilingual text (millions of sentence pairs or more), the system can recognize words and phrases from the input sentence and recall how humans have translated these pieces in the past. Using a collection of statistical models, the system can predict the most likely human translation of the new sentence given what it has seen before. Recognizing larger pieces of an input sentence leads to better translation quality; in the best case, an entire sentence can be recognized and a human quality translation can be recalled while in the worst case, each word must be recalled from a different translation and pieced together. This puts a vital importance on the availability of data that is similar to what needs to be translated. Current statistical machine translation systems use a range of techniques to learn ideal units of translation from such data, match them against unseen source sentences, and piece their translations together in a way that seems reasonable in the target language. In this section, we build up to the hierarchical phrase-based approach to machine translation that we use in our work. For further reading on the motivation for and theory of statistical machine translation, see Kevin Knight's tutorial on earlier word-based translation (1999) and Chapter 1 of Adam Lopez's dissertation (2008a) that surveys more recent work.

1.1.1 Word Alignment

Early statistical translation models (Brown et al., 1993) are word-based, explaining translation as the following lexical process. For a given source language sentence $F = \langle f_1, ..., f_n \rangle$ with length n, generate a target sentence length m and set of alignment links $A = \langle a_1, ..., a_m \rangle$. Finally, generate a target sentence $E = \langle e_1, ..., e_m \rangle$ where each target word e_i only depends on the source word f_{a_i} that it is aligned to. Current alignment models handle differences in length between source and target sentences by allowing each f_i to generate any number of links, aligning to zero or more words e_i. Target words e_i can similarly align to zero or more words f_i. Differences in word order between source and target languages are accounted for by allowing alignment links to be unordered with respect to the source sentence. For example, Figure 1.1 shows an alignment where f_2 (de) is aligned to both e_1 and e_3, and f_1 (devis), is aligned to e_4.

While modern systems do not translate with word based models directly, the intermediate alignments between source and target words produced by these simple models are still useful as a starting point for

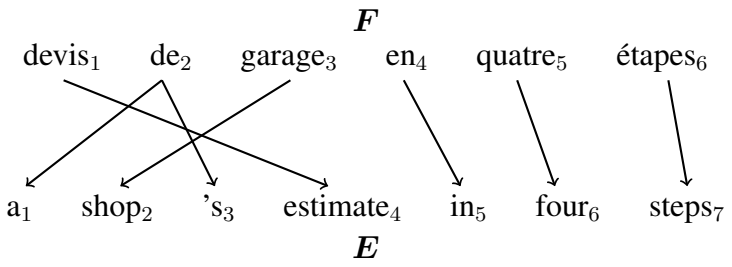


Figure 1.1: Visualization of French-to-English word alignment with one-to-many alignments and reordering

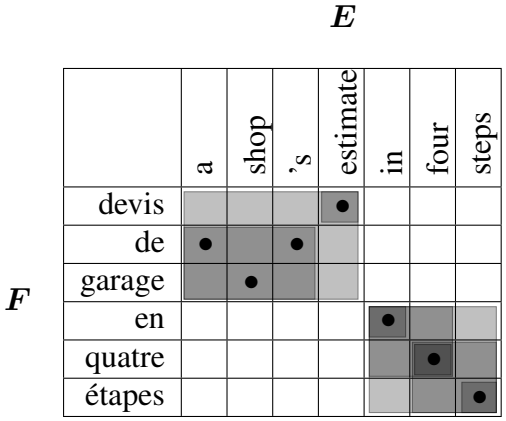


Figure 1.2: Visualization of phrase extraction from an aligned sentence pair. Dots signify alignment links and shaded boxes signify extracted phrases. Phrase length is limited to 3 words on the source side. A total of 9 phrases are extracted.

building more sophisticated models. One shortcoming of these alignments is that since they come from directional models, they are unable to map multiple source words to a single target word, or multiple source words to multiple target words. To account for this, models are typically run in both directions (source-to-target and target-to-source) and the alignments symmetrized (Och and Ney, 2003; Axelrod et al., 2005). Symmetrization uses information from both alignments to produce a single, bidirectional alignment that supports one-to-many alignments in either direction.

1.1.2 Bilingual Phrase Extraction

Individually translating each word in a source sentence without context and permuting the result into something meaningful is clearly problematic. Once bilingual text has been aligned at the word level, phrase-based models (Koehn et al., 2003; Och and Ney, 2004; Och et al., 1999) can learn more reliable mappings between source and target languages by grouping together sequences of words into atomic units of translation. For example, rather than relying on the complex alignment process in Figure 1.1 to translate "devis de garage", a phrase-based model can simply learn that the whole phrase translates into "a shop's estimate". These mappings, termed *phrase pairs*, can be extracted automatically from word-aligned text. Given an aligned source-target sentence pair $\langle F, E, A \rangle$, phrase pairs consistent with the alignment can be identified as follows. A phrase pair $\langle f_i^{i+n}, e_j^{j+m} \rangle$ covering the contiguous span of words from i to $i+n$ in the source sentence

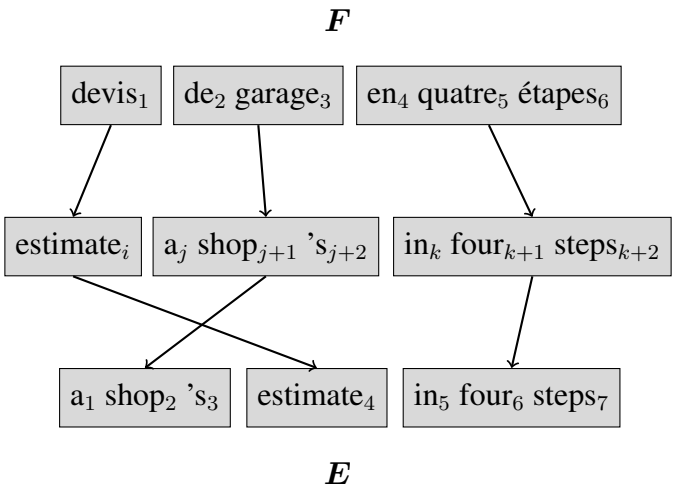


Figure 1.3: Visualization of phrase-based segmentation, translation, and reordering with 3 phrase pairs

and from j to $j+m$ in the target sentence is extracted if (1) at least one word in f_i^{i+n} is aligned to a word in e_j^{j+m} and (2) no word in f_i^{i+n} is aligned to any word outside e_j^{j+m} and vice versa. Formally, the bilingual phrase pairs for a given sentence pair are defined:

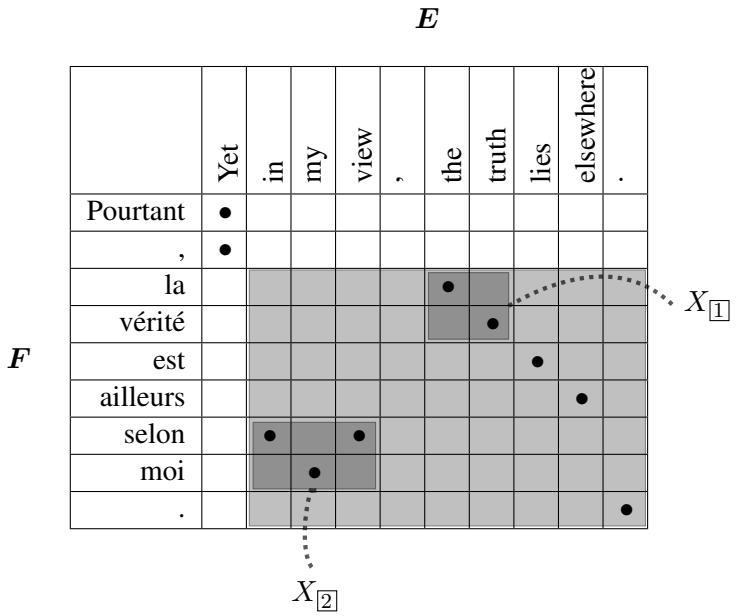
$$\begin{aligned}\texttt{BPP}(F,E,A) = \Big\{ \langle f_i^{i+n}, e_j^{j+m} \rangle \;\Big|\; & \forall \langle i', j' \rangle \in A : i \leq i' \leq i+n \iff j \leq j' \leq j+m \\ & \wedge \exists \langle i', j' \rangle \in A : i \leq i' \leq i+n \wedge j \leq j' \leq j+m \Big\}\end{aligned} \tag{1.1}$$

An example of phrase extraction is visualized in Figure 1.2. Note that many overlapping phrases can be extracted from the same sentence pair.

Once phrases are learned, the task of translating a new source sentence F consists of decomposing it into a series of phrases, rewriting each phrase with its target language equivalent, and permuting the order of phrases on the target side to produce the final sentence E. Translating at the phrase level has the key advantages of context and encapsulation. While individual words can have many translations, longer phrases are generally less ambiguous. While a word-based model cannot distinguish between the French preposition "en" and the English language code abbreviation "en", a phrase-based model can match the longer phrase "en quatre étapes", using additional context to resolve translation ambiguity. Other phrases such as "devis de garage" that would require complicated word mapping and reordering in word-based translation can be captured in a single phrase pair. Other complex natural language phenomena such as idiomatic phrases and morphological inflection can also be encapsulated in phrase pairs, allowing for a single phrase rewrite operation to generate a human quality translation for difficult content. However, this also underscores the importance of having seen at least one instance of a given language construction in the training text.

1.1.3 Hierarchical Phrase Translation Models

While phrase-based models excel at translating series of short, self-contained phrases, longer sentences pose significant challenges. Language phenomena such as long distance reordering and word agreement require context beyond what can be easily encapsulated in short phrases. To correctly translate long, complex sentences, phrase-based systems must make sequences of unintuitive independent translation and reordering decisions reminiscent of word-based models. This problem is addressed in the hierarchical phrase-based



$$X \longrightarrow X_{\boxed{1}} \text{ est ailleurs } X_{\boxed{2}} \text{ . } / \; X_{\boxed{2}} \text{ , } X_{\boxed{1}} \text{ lies elsewhere .}$$

Figure 1.4: Visualization of a hierarchical phrase pair extracted from an aligned sentence pair. The linked non-terminals in the resulting SCFG rule encode reordering between the source and target.

formalism (Chiang, 2007) by introducing hierarchical phrases that can contain other phrases, allowing long distance context to be encapsulated in the same way as local context. With generalized phrases, larger portions of text can be grouped together and reordered within the context of a single phrase pair. Given an aligned source-target sentence pair $\langle F, E, A \rangle$, hierarchical phrase pairs can be extracted as follows. First, identify initial phrase pairs that meet the criteria in Equation 1.1. Next, identify phrases that contain other phrases and replace the source and target words covered by each sub-phrase with a special indexed symbol $X_{\boxed{i}}$. These symbols indicate where other phrase pairs can be plugged in. To keep the number of extracted rules manageable, the following additional constraints are imposed: (1) phrases and sub-phrases must be *tight*, meaning that boundary words must be aligned, (2) there must be at least one word between any two $X_{\boxed{i}}$ symbols in the source phrase, (3) there must be at least one word in the source phrase, and (4) there may be at most two $X_{\boxed{i}}$ symbols in any phrase. An instance of hierarchical phrase extraction is visualized in Figure 1.4. Note this is just one of many hierarchical phrase pairs could be extracted from the example sentence pair.

Under this model, phrase pairs can also be expressed as rules in a synchronous context-free grammar (SCFG) where all source and target phrases are given the same label X. Formally, any translation *rule* extracted from data can be written:

$$X \longrightarrow \bar{f} \; / \; \bar{e} \tag{1.2}$$

Here $\bar{f}$ denotes a source-language phrase and $\bar{e}$ denotes a target-language phrase. Phrases *must* contain words and *may* contain linked non-terminals $X_{\boxed{i}}$ (see example rule in Figure 1.4). The task of translation is now equated to parsing the source sentence with the translation grammar, simultaneously building up a target-language derivation and ultimate translation. To allow building full derivations, a single goal non-terminal S is added to the translation grammar. To facilitate the phrase-based approach of dividing

$\boldsymbol{F}$: Pourtant , la vérité est ailleurs selon moi .

$\boldsymbol{G}$:

$S \longrightarrow S_{\boxed{1}}\ X_{\boxed{2}} \ / \ S_{\boxed{1}}\ X_{\boxed{2}}$

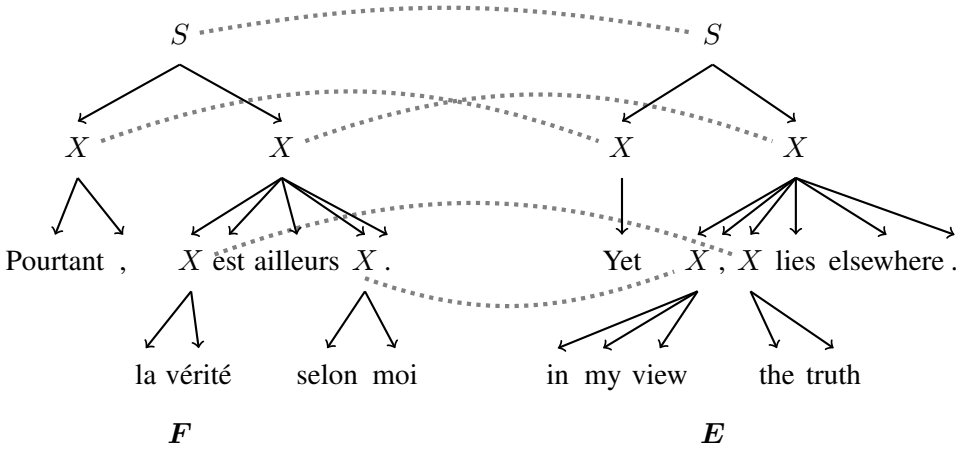
$S \longrightarrow X_{\boxed{1}} \ / \ X_{\boxed{2}}$

$X \longrightarrow X_{\boxed{1}}$ est ailleurs $X_{\boxed{2}}$. $/$ $X_{\boxed{2}}$, $X_{\boxed{1}}$ lies elsewhere .

$X \longrightarrow$ Pourtant , $/$ Yet

$X \longrightarrow$ la vérité $/$ the truth

$X \longrightarrow$ selon moi $/$ in my view



$\boldsymbol{S_{\boxed{1}}} \ / \ \boldsymbol{S_{\boxed{1}}}$

$\Longrightarrow \quad \boldsymbol{S_{\boxed{2}}}\ \boldsymbol{X_{\boxed{3}}} \ / \ \boldsymbol{S_{\boxed{2}}}\ \boldsymbol{X_{\boxed{3}}}$

$\Longrightarrow \quad \boldsymbol{X_{\boxed{4}}}\ X_{\boxed{3}} \ / \ \boldsymbol{X_{\boxed{4}}}\ X_{\boxed{3}}$

$\Longrightarrow \quad$ **Pourtant ,** $X_{\boxed{3}}$ $/$ **Yet** $X_{\boxed{3}}$

$\Longrightarrow \quad$ Pourtant , $\boldsymbol{X_{\boxed{5}}}$ **est ailleurs** $\boldsymbol{X_{\boxed{6}}}$ **.** $/$ Yet $\boldsymbol{X_{\boxed{6}}}$ **,** $\boldsymbol{X_{\boxed{5}}}$ **lies elsewhere .**

$\Longrightarrow \quad$ Pourtant , **la vérité** est ailleurs $X_{\boxed{6}}$. $/$ Yet $X_{\boxed{6}}$, **the truth** lies elsewhere .

$\Longrightarrow \quad$ Pourtant , la vérité est ailleurs **selon moi** . $/$ Yet **in my view** , the truth lies elsewhere .

Figure 1.5: Example of translation as parsing with a synchronous context-free grammar. Top: F is a sample source language sentence and G is a sample translation grammar. Center: visualization of the trees generated from parsing F with G, also building a target sentence E. Dotted lines indicate that non-terminals share indices. Bottom: synchronous derivation of F and E under G.

input sentences into individually-translatable chunks, two *glue rules* are added that string together series of phrases:

$$\begin{aligned} S &\longrightarrow S_{\boxed{1}}\, X_{\boxed{2}} \;/\; S_{\boxed{1}}\, X_{\boxed{2}} \\ S &\longrightarrow X_{\boxed{1}} \;/\; X_{\boxed{2}} \end{aligned} \tag{1.3}$$

Figure 1.5 shows an example of translation as parsing with a synchronous context-free grammar.

In addition to providing a powerful generalization of the phrase-based formalism, the hierarchical approach can be considered an unsupervised version of syntactic machine translation. In syntactic translation, models learn correspondences between source and target language *structure* from bilingual text that has been annotated with parse trees (sometimes called concrete syntax trees) on the source (Yamada and Knight, 2001; Liu et al., 2006), the target (Galley et al., 2004), or both (Lavie et al., 2008; Liu et al., 2009) sides. This additional information allows syntactic models to learn SCFG rules by dividing source and target parse trees into corresponding chunks based on word-level alignments, then translate new sentences by parsing them with the resulting translation grammar. While the hierarchical phrase-based model does not have access to parse information, it can learn similar translation rules based solely on word alignments. The end result is a model that incorporates strengths of syntactic approaches while retaining the flexibility of a phrase-based model. This is the translation formalism used throughout our work.

1.2 Translation Model Parameterization

In the previous section, we formally introduced the hierarchical phrase-based translation model and described the process for extracting translation rules from bilingual text and applying them to translate unseen sentences. The performance of these models is highly dependent on the amount of training data available, with models for high-traffic language pairs such as Spanish–English and Arabic–English typically being learned from millions of bilingual sentence pairs. Given the natural ambiguity of human language and the need to piece together translations from such large numbers of sources to translate new content, current machine translation systems employ several statistical models to predict the single *most likely* translation of a source sentence given all data the system has seen previously. This amounts to scoring and ranking the often exponential number of possible translation candidates for a single sentence. This process is referred to as *decoding* and the programs that conduct this process *decoders*. In this section, we describe decoding (inference) and the process of estimating the prerequisite translation models from bilingual text (learning).

1.2.1 Linear Translation Models

The translation model described in §1.1.3 can predict exponentially many translations for each source sentence. The model needs a way to score each possible translation so that it can select the single most likely candidate. The dominant approach, which we use throughout our work, is a translation model parameterization by Och and Ney (2002; 2003). A translation *hypothesis* consists of F, the input source-language sentence, D, the derivation (collection of SCFG rules) that maps F to some target-language sentence E, and E itself, the translation we are ultimately interested in. We then introduce arbitrary *feature functions* $\mathcal{H}_i \in \mathcal{H}$ into our model that assign real-number values to hypotheses. Further described in §1.2.2 and §1.2.4, these functions typically measure how reliably F translates into E or how well-formed of a sentence E is in the target language. For each $\mathcal{H}_i$, a corresponding *weight* $w_i \in W$ controls the relative contribution of the feature to the final score, allowing the model to trust some features more than others. Setting these weights (collectively called a weight vector) to maximize system performance is discussed in §1.3. By calculating the inner product of feature scores and weights for a given translation, one obtains a final score that can be compared against scores of other translations. Formally, the score of a translation hypothesis $\langle F, D, E \rangle$ can

be written:

$$S(F, D, E) = \sum_{i=1}^{|\mathcal{H}|} w_i \mathcal{H}_i(F, D, E) \tag{1.4}$$

This leads to the translation decision rule to select $\hat{E}$, the target language sentence with the highest score under the model:

$$\hat{E}(F) = \arg\max_{\langle E,D \rangle} \sum_{i=1}^{|\mathcal{H}|} w_i \mathcal{H}_i(F, D, E) \tag{1.5}$$

The decision rule, used directly by our model, is a straightforward formulation of hypothesis score as the inner product of a feature score vector and a and feature weight vector. As observed by Clark (2012), when a translation model uses only this decision rule with arbitrary feature functions and weights, the model is linear rather than log-linear. The model is highly extensible and facilitates learning weights so as to directly maximize translation quality on held-out data.

1.2.2 Rule-Local Features

To score translation hypotheses, we add real-valued *local* features h_i to each rule in the translation grammar, making it a weighted SCFG. The *global* value of each feature function (used in Equations 1.4 and 1.5) is the sum of the local features used in the derivation:

$$\mathcal{H}_i(D) = \sum_{X \to \bar{f}/\bar{e} \in D} h_i\left(X \to \bar{f}/\bar{e}\right) \tag{1.6}$$

By assuming rules have the same feature values independent of context, efficient inference is possible with dynamic programming. While the scores $h_i(X \to \bar{f}/\bar{e})$ assigned to each rule can be arbitrary, they generally reflect how consistent a translation rule is with bilingual training data or provide other information about the current derivation during model search (search is further discussed in §1.2.4).

Phrase Features: Given a rule $X \to \bar{f}/\bar{e}$, these features encode the empirical relative frequency of a given source phrase $\bar{f}$ being translated as a target phrase $\bar{e}$ according to the bilingual training data. Here the training data is the set of all rule instances extracted from all sentences in the training text. Counting rules that share source, target, and both sides leads to the following statistics:

- $\mathcal{C}(\bar{f}, \bar{e})$: the count of times the rule with source $\bar{f}$ and target $\bar{e}$ is extracted.
- $\mathcal{C}(\bar{f})$: the count of times any rule with source $\bar{f}$ is extracted with any target.
- $\mathcal{C}(\bar{e})$: the count of times any rule with target $\bar{e}$ is extracted with any source.

Given these statistics, two standard feature scores are calculated:

$$f(\bar{e}|\bar{f}) = \frac{\mathcal{C}(\bar{f}, \bar{e})}{\mathcal{C}(\bar{f})} \qquad f(\bar{f}|\bar{e}) = \frac{\mathcal{C}(\bar{f}, \bar{e})}{\mathcal{C}(\bar{e})} \tag{1.7}$$

Lexical Features: Since individual words generally occur far more frequently than whole phrases, word-level translation scores can be effectively estimated from much larger data, leading to more reliable estimates. Adding these *lexical* scores to the linear translation model can be seen as smoothing the less stable phrase-based translation scores with word-based translation scores. Given a rule $X \to \bar{f}/\bar{e}$, lexical features encode the probability of the words $e \in \bar{e}$ being *individually* mapped to the words $f \in \bar{f}$. Here the training data is the set of all alignment links from all word alignments in the bilingual training text. Counting instances of aligned words leads to the following statistics:

- $\mathcal{C}(f, e)$: the count of times source word f is aligned to target word e. When counting links, one-to-many alignments are accounted for by adding a fractional count of $\frac{1}{n}$ for any instance where f or e is aligned to n words instead of one.
- $\mathcal{C}(f)$: the count of times source word f is aligned to one or more target words.
- $\mathcal{C}(e)$: the count of times target word e is aligned to one or more source words.

Word-level lexical scores f_w are calculated:

$$f_w(e|f) = \frac{\mathcal{C}(f, e)}{\mathcal{C}(f)} \qquad f_w(f|e) = \frac{\mathcal{C}(f, e)}{\mathcal{C}(e)} \tag{1.8}$$

These scores are then used to calculate phrase-level lexical scores f_{lex}. When aligning words within phrases, we use an approximation wherein each source word $f \in \bar{f}$ is aligned to the target word $e \in \bar{e}$ with the highest word-level score. Formally, the two lexical scores are calculated:

$$f_{\text{lex}}(\bar{f}|\bar{e}) = \prod_{f \in \bar{f}} \arg\max_{e \in \bar{e}} f_w(f|e) \qquad P_{\text{lex}}(\bar{e}|\bar{f}) = \prod_{e \in \bar{e}} \arg\max_{f \in \bar{f}} f_w(e|f) \tag{1.9}$$

We include the log-transformed[1] versions of these features in our model:

$$\texttt{MaxLex(f|e)} = \log f_{\text{lex}}(\bar{f}|\bar{e}) \qquad \texttt{MaxLex(e|f)} = \log f_{\text{lex}}(\bar{e}|\bar{f}) \tag{1.10}$$

1.2.3 On-Demand Grammar Extraction and Suffix Array Features

Traditionally, the entire training data is used to extract sufficient statistics and estimate a single weighted SCFG prior to translating any input sentences. In a development by Callison-Burch et al. (2005) and Lopez (2008a; 2008b), the training data can instead be indexed using a suffix array (Manber and Myers, 1993), a data structure optimized for fast text string lookups, allowing the estimation of sentence-specific translation grammars as needed. When an input sentence needs to be translated, a grammar extraction program can find all possible decompositions of the source sentence into phrases and search for these phrases in the suffix array. The suffix array returns instances of each source phrase in the training data, along with the target-language phrases the source phrase is aligned to. This data can be used to score a sentence-level grammar that translates the input sentence. Instead of using all occurrences of each source phrase in the data (potentially millions for common phrases), a smaller *sample* can be used to estimate feature scores. Lopez (2008a) finds that a sample size of 100 performs comparably to using the entire data. This allows for both the rapid generation of grammars on an as-needed basis and the inclusion of a powerful suffix array-backed feature set, discussed below.

Suffix Array Phrase Features: A source-side suffix array provides valuable information that can be used to estimate a more powerful set of phrase features. For each source phrase $\bar{f}$, the suffix array returns a sample $\mathcal{S}$ that consists of instances $\langle \bar{f}, \bar{e}' \rangle$ where $\bar{e}'$ is the target-language phrase that $\bar{f}$ is aligned to in the given instance. In the case that $\bar{f}$ is unaligned, $\bar{e}'$ is empty and no rule can be instantiated. A single $\mathcal{S}$ is used to score all rules $X \rightarrow \bar{f}/\bar{e}'$ that can be instantiated over $\bar{f}$ and $\bar{e}'$. Feature scores are calculated using the following statistics:

- $\mathcal{C}_{\mathcal{S}}(\bar{f}, \bar{e})$: the count of instances in $\mathcal{S}$ where $\bar{f}$ is aligned to $\bar{e}$. Also called the co-occurrence count, we use the log-transformed version of this statistic as a feature score in our model:

$$\texttt{Count(f,e)} = \log \mathcal{C}_{\mathcal{S}}(\bar{f}, \bar{e}) \tag{1.11}$$

[1] While our translation model is linear rather than log-linear, the conventional log transformation still provides performance improvements for probability and count-based features.

- $\mathcal{C}_{\mathcal{S}}(\bar{f})$: the count of instances in $\mathcal{S}$ where $\bar{f}$ is aligned to any target phrase.
- $|\mathcal{S}|$: the total number of instances in $\mathcal{S}$, equal to the number of occurrences of $\bar{f}$ in the training data, up to the sample size limit. We use the log-transformed version of this statistic as a feature score in our model:

$$\texttt{SampleCount(f)} = \log |\mathcal{S}| \tag{1.12}$$

In addition to the above statistics, we incorporate two *indicator* features into our model that return a value of one if their conditions are met, otherwise zero. These features are used to count rare singleton translations where only one instance of $\bar{f}$ or $\langle \bar{f}, \bar{e} \rangle$ exist in the training data. Given the above statistics, we add the following two features to our model:

$$\texttt{Singleton(f)} = \begin{cases} 1 & \mathcal{C}_{\mathcal{S}}(\bar{f}) = 1 \\ 0 & \text{otherwise} \end{cases} \qquad \texttt{Singleton(f,e)} = \begin{cases} 1 & \mathcal{C}_{\mathcal{S}}(\bar{f}, \bar{e}) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1.13}$$

Finally, the suffix array index of all instances of $\bar{f}$ in the source side of the training data allows for the calculation of a more powerful phrase translation score. Termed the *coherent* translation score, this variant conditions on the frequency of $\bar{f}$ in the data rather than frequency of $\bar{f}$ being extracted as part of a phrase pair. The formula and subsequent log-transformed feature function used in our model are given:

$$\text{coherent } P(\bar{e}|\bar{f}) = \frac{\mathcal{C}_{\mathcal{S}}(\bar{f}, \bar{e})}{|\mathcal{S}|} \qquad \texttt{CoherentP(e|f)} = \log \text{coherent } P(\bar{e}|\bar{f}) \tag{1.14}$$

The use of $|\mathcal{S}|$ instead of $\mathcal{C}_{\mathcal{S}}(\bar{f})$ increases the value of the denominator in cases where the source phrase is frequently unaligned, preventing rule extraction. This reduces the feature value of rules for which the source side tends not to align well.

1.2.4 Additional Features

Language Model Features: In addition to feature scores assigned to SCFG rules that encode the likelihood of source phrases translating into target phrases, a machine translation systems employs a *language model* that assigns scores to the target language sentence $E = \langle e_1 ... e_{|E|} \rangle$. Language model scores reflect $P(E)$, the likelihood of sentence E occurring given the monolingual training text. Linguistically, language models can be seen as a measure of grammaticality and fluency, how well formed the translation hypothesis is in the target language. Standard language models use an N-gram approximation where the probability of a word e_i is conditioned on the previous N-1 words, typically 3 or 4. Since the model matches E against the training data, this approximation greatly reduces sparsity; entire sentences that the translation model generates are unlikely to appear in training data, but short sequences of words are more likely. Formally, the probability under an N-gram language model and the corresponding log-transformed feature in our system are given:

$$P_N(E) = \prod_{i=1}^{|E|} P_N(e_i|e_1^{i-1}) = \prod_{i=1}^{|E|} P_N(e_i|e_{i-N}^{i-1}) \qquad \texttt{LM(E)} = \log P_N(E) \tag{1.15}$$

N-gram probabilities for language models use smoothed maximum likelihood estimates on the training data, which consists of all N-gram instances that occur in the monolingual text. When an N-gram is not found, rather than assigning zero probability, the model can "back off" to the probability for a shorter context, starting with N-2, down to 0. If the current word is not in the vocabulary of the model, a single probability

for out-of-vocabulary (OOV) words is applied. To fine-tune the impact of OOV words, an additional count-based feature is added to track the number of OOV words in E:

$$\texttt{OOV(E)} = \sum_{e \in E} \begin{cases} 1 & e \text{ in language model} \\ 0 & \text{otherwise} \end{cases} \tag{1.16}$$

As monolingual data is generally far more plentiful than bilingual data, language models are estimated from orders of magnitude more data than translation models, making language model scores powerful discriminative features. It is important to note that translation (phrase and lexical) features and language model features operate independently within the linear model, each weighing in on the quality of a translation hypothesis. Hypotheses with higher $P(E|F)$ frequently have higher $P(E)$ but this is not always the case, especially when translating out-of-domain text where bilingual and monolingual training data may be mismatched.

Derivation Features: The final set of features encodes information about the derivation D being built as the source sentence is parsed with the translation grammar. These features allow the optimizer to learn to prefer characteristics of derivations that tend to lead to good translations. The optimal weights for these features can be highly dependent on language pair:

- `Arity(D,0)`: the number of rules with zero non-terminals $X_{\boxed{i}}$ used in D.
- `Arity(D,1)`: the number of rules with one non-terminal $X_{\boxed{i}}$ used in D.
- `Arity(D,2)`: the number of rules with two non-terminals $X_{\boxed{i}}$ used in D.
- `GlueCount(D)`: the number of glue rules (Equation 1.3) used in D.
- `PassThroughCount(D)`: the number of out-of-vocabulary source words passed through in D.
- `WordCount(E)`: the number of words in the translation hypothesis E. This feature directly counterbalances the bias toward shorter sentences favored by the language model. Multiplying language model probabilities for additional words continues to lower the final score, leading the language model to naturally prefer shorter sentences even when the individual N-grams receive low scores. A positively weighted word count feature can reward longer translations, leading to a better balance between translation quality and length.

1.3 Translation System Optimization

One of the key advantages of the linear translation model discussed in §1.2.1 is the ability to add any real-valued feature function and assign a weight to control its contribution to scoring hypotheses. This theoretically allows any knowledge source to be adapted to provide information to a translation model, relying on the optimizer to decide how useful it is. Learning ideal weights for these features is an important problem. Translation system optimization is generally formulated as choosing the weight vector that, when used with a fixed set of features, is most likely to result in good translations of future text. This is complicated by (1) the lack of a clear definition of translation goodness, (2) large search spaces, and (3) the difficulty many algorithms have with finding optimal weights for large numbers of often correlated feature functions. To address the first point, automatic evaluation metrics are introduced that compare translation hypotheses against pre-generated human translations and generate a similarity score (frequently called distance) in a well-defined way. To address the second, newer optimization algorithms use different objective functions intended to scale to much larger feature sets. In this section, we describe two popular translation system optimization techniques and the metrics they use to score translations.

1.3.1 Minimum Error Rate Training

One widely used method for learning feature weights is minimum error rate training (MERT) (Och, 2003), which directly optimizes system performance on a given development data set (also called a tuning set). The intuition here is that the system parameters that lead to the best translations for known data are likely to lead to good translations for unknown data. Translation quality is measured by an automatic metric $\mathcal{G}$ that returns a similarity score between a system's output E' and a human reference translation E at the corpus level. MERT searches for the weight vector $\hat{W}$ that leads the translation model to prefer the E' closest to E, maximizing the metric score. Given a bilingual development corpus C that consists of sentence pairs $\langle F, E \rangle$, MERT's optimization function is given:

$$\hat{W} = \arg\max_{W} \sum_{\langle F,E \rangle \in C} \mathcal{G}\left(\hat{E}(F), E\right) \tag{1.17}$$

Using Equation 1.5, we expand the translation model's decision rule $\hat{E}(F)$ to show the direct impact of feature weights on translation selection and consequently score:

$$\hat{W} = \arg\max_{W} \sum_{\langle F,E \rangle \in C} \mathcal{G}\left(\arg\max_{\langle E',D' \rangle} \sum_{i=1}^{|\mathcal{H}|} w_i h_i(F, E', D'), E\right) \tag{1.18}$$

The search for $\hat{W}$ proceeds as an iterative line search. First, the translation system uses an initial set of weights (uniform, random, or set based on some amount of prior knowledge) to translate the source sentences in C. Rather than producing the single most likely translation under the model, the system outputs the K most likely translations, either as a list or packed into a lattice or hypergraph (Macherey et al., 2008). Candidates in the K-best list are annotated with feature scores, allowing them to be re-scored with different weight vectors. A weight vector W is scored by calculating the metric score for the single candidate in each K-best list preferred by the model under W and aggregating the results. MERT then conducts a sequence of individual searches, finding the best-scoring value for each weight $w \in W$ while holding the values of all other weights fixed. Once MERT has optimized the entire weight set, the system generates a set of K-best lists with the new weight vector and aggregates them to lists from previous iterations. This allows the line search to view increasingly large portions of the space of possible translations under the model. In addition to searching from the best W in the previous iteration, MERT also searches several random weight vectors in each iteration to reduce the chance of getting stuck in a local optimum. MERT concludes when no new translations are discovered under the current weight vector W, selecting the final vector $\hat{W}$ that results in the best known score given the visible space of translations.

While MERT is still the de facto optimization algorithm for translation systems, its design leads to a few natural drawbacks. Only looking at the metric score of the top-best translation for each sentence in a development set can lead MERT to prefer local optima that do not generalize well to other data sets; the optimizer may select weights that are overly specific to C at the expensive of performance on other data, *overfitting* the tuning set. Additionally, the line search method does not scale well to larger feature sets, especially when features are correlated; searching one parameter at a time can miss complex interactions between features. For sets of more than a few dozen features, line search actually becomes intractible.

1.3.2 Pairwise Rank Optimization

To address several shortcomings of MERT, Hopkins and May (2011) introduce the pairwise rank optimization (PRO) algorithm for tuning translation systems. PRO aims to learn better-generalizing weight vectors by focusing on a system's ability to discriminate between good and bad translations rather than maximizing the score of a single set of top-best translations for a development set. Shifting to a ranking approach

also allows the parameter search to be recast as binary classification, a well established problem that can be solved with existing tools and scaled to thousands of features. Using the same type of development set C that includes sentence pairs $\langle F, E \rangle$, PRO optimizes in an iterative fashion similar to MERT. For each iteration, the system generates a K-best hypothesis list for each sentence, aggregating it with lists of previous iterations to increase the optimizer's view of the translation space. The optimizer then finds the best weight vector given the visible translation space and proceeds to the next iteration.

The optimization problem solved during each iteration of PRO can be formulated as follows. For each source sentence F, randomly sample Γ pairs of translations $\langle E'_i, E'_j \rangle$ from the K-best list. Determine the difference between E'_i and E'_j by scoring them independently against the reference translation E with an automatic metric $\mathcal{G}$ and subtracting their scores. Only accept pairs with a absolute difference greater than some threshold α to avoid having to choose between two nearly-identical translations as is often the case in MERT's top-best optimization. Choose the top Ξ pairs out of Γ with the largest score difference and determine their relationship:

$$E'_i > E'_j \iff \mathcal{G}(E'_i, E) - \mathcal{G}(E'_j, E) > 0 \tag{1.19}$$

For each pair, generate a positive and negative training instance based on the inequality. For example, if $E'_i > E'_j$, generate $\langle E'_i, E'_j, + \rangle$ and $\langle E'_j, E'_i, - \rangle$. Pool the resulting 2Ξ training instances with those from all other sentences in the tuning corpus to form the training data for binary classification. Finally, run a standard binary classifier (Hopkins and May use MegaM (Daumé III, 2004)) to find the weight vector $\hat{W}$ that leads to model scores that correctly rank the largest number of instances in the training data. Unlike MERT, PRO has no inherent stopping criteria, so it is typically run for a fixed number of iterations.

1.3.3 Evaluation Metrics

Ideally, a translation's quality (human or automatic) should be measured by its usefulness to humans, either for information assimilation or as a starting point for post-editing. In practice, human evaluation is often infeasible as evaluations need to be carried out rapidly and repeatedly. Automatic evaluation metrics simulate human judgments by comparing a translation hypothesis against a pre-existing reference translation and return a numerical similarity score, generally between zero and one. The role of metrics in system development is twofold. First, metrics are used to provide the thousands of individual evaluations required in algorithms like MERT and PRO. Second, metrics are used to score the output of optimized systems on held out data to determine which system or system configuration performs best. This section describes three metrics frequently used for optimization and evaluation: BLEU, TER, and Meteor.

BLEU: Based on the idea that good translations should contain words and phrases from references, the bilingual evaluation understudy (BLEU) metric (Papineni et al., 2002) scores hypotheses according to surface form N-gram precision. For every n-gram length up to N, ($N = 4$ in the widely used BLEU_4 variant), an individual precision score $\mathcal{P}_n$ is calculated as the percentage of n-grams in the hypothesis also found in the reference. Precision scores are combined using a geometric mean and scaled by a brevity penalty intended to down-weight hypotheses that achieve good precision but are too short to achieve good recall. The penalty ($\mathcal{B}$) is based on the length of the translation hypothesis E' and reference E. The formula for BLEU score is given:

$$\texttt{BP}(E', E) = \begin{cases} 1 & \text{if } |E'| > |E| \\ e^{\frac{1-|E|}{|E'|}} & \text{if } |E'| \leq |E| \end{cases} \qquad \texttt{BLEU}_N(E', E) = \texttt{BP} \times \exp\left(\sum_{n=1}^{N} \frac{1}{N} \log \mathcal{P}_n\right) \tag{1.20}$$

To account for translation variation, multiple reference translations can be used to score a single hypothesis. N-grams from the hypothesis can match any reference, but the same N-gram cannot be matched more times

than it occurs in any one reference. The reference length closest to the hypothesis length is used to calculate the brevity penalty $\mathcal{B}$. BLEU scores range from 0 to 1 and are often reported as percentages (e.g., 27.3 BLEU "points" for a score of 0.273).

While BLEU is still the dominant metric for optimization and evaluation, it is frequently criticized for being insensitive to important translation differences. Callison-Burch et al. (2006) show that improvement in BLEU score is neither necessary nor sufficient for improvement in translation quality as assessed by humans. BLEU ignores linguistic phenomena such as synonymy and paraphrasing, penalizing translations that use different vocabulary and phrasing to express the same meaning as reference translations. As N-grams can be matched anywhere in a hypothesis, BLEU is also insensitive to non-local ordering, unable to discriminate between globally coherent sentences and scrambled sentences. The end result is that many diverse sentences in K-best lists appear identical to BLEU, leading to unreliable feedback for optimization algorithms such as MERT. This also motivates algorithms such as PRO (§1.3.2) to focus on ranking cases with large differences in metric score to minimize the impact of this unreliability.

TER: The translation edit rate (TER) metric (Snover et al., 2006) is based on the idea that good translations should require minimal effort to correct. TER defines four basic operations that can be used to edit hypotheses: single word insertion, single word deletion, single word substitution, and block shift, wherein a contiguous span of words is moved as a single unit. TER is defined as the *minimum* number of equally-weighted edit operations $\mathcal{C}_{\texttt{edit}}$ required to transform a translation hypothesis E' into a reference E, normalized by the length of the reference. In the case of multiple references, TER is defined as the minimum edit distance over all references, normalized by the *average* reference length. Formally:

$$\texttt{TER}(E', E) = \frac{\mathcal{C}_{\texttt{edit}}(E', E)}{|E|} \tag{1.21}$$

TER scores range from 0 to infinity, though in practice they are capped at 1. While much less widely used than BLEU for optimization and evaluation, TER plays a key role as an approximation of human post-editing effort, discussed in detail in §1.4.2. As TER is an *error* measure, *lower* scores are better.

Meteor: Engineered to address weaknesses of previous metrics, Meteor (Banerjee and Lavie, 2005; Lavie and Denkowski, 2009) is based on the idea that good translations should *align* well to references, just as source sentences align to equivalent target sentences. The Meteor aligner introduces flexible word matching to account for translation variation. When evaluating a hypothesis E' against a reference E, Meteor creates a word alignment based on exact (surface form), stem, and synonym matches. The total number of word matches h in the hypothesis and reference are used to calculate precision and recall:

$$\mathcal{P} = \frac{h(E')}{|E'|} \qquad \mathcal{R} = \frac{h(E)}{|E|} \tag{1.22}$$

$\mathcal{P}$ and $\mathcal{R}$ are combined in a weighted harmonic mean that scores word choice:

$$\mathcal{F}_\alpha = \frac{\mathcal{P} \times \mathcal{R}}{\alpha \times \mathcal{P} + (1 - \alpha) \times \mathcal{R}} \tag{1.23}$$

The number of chunks, (Ch) is calculated as the minimum number of contiguously aligned word spans the alignment can be divided into. A fragmentation score (Frag) assesses word order and global coherence:

$$\texttt{Frag} = \frac{\texttt{Ch}}{h(E')} \tag{1.24}$$

A final sentence-level score is calculated:

$$\texttt{Meteor}(E', E) = \left(1 - \gamma \times \texttt{Frag}^\beta\right) \times \mathcal{F}_\alpha \tag{1.25}$$

Three tunable parameters (α, β, and γ) allow adjusting the relative importance of precision, recall, and reordering. Agarwal and Lavie (2008) tune these parameters to maximize correlation with human judgments of translation quality, leading to greatly improved accuracy over BLEU and TER. Despite higher correlation with human assessments, Meteor is typically not used to tune translation systems.

1.4 Human and Machine Translation

While automatic translation is sometimes sufficient for conveying information across language barriers, many scenarios still require high quality human translation. Governments and international organizations such as the United Nations require accurate translations of content dealing with complex geopolitical issues. Businesses require localization that preserves the image of goods and services. Community-driven projects such as Wikipedia[2] (Wikipedia, 2013) rely on volunteer translators to bring information and resources to diverse language communities. As the amount of data requiring translation has continued to increase, the idea of using machine translation to improve the speed of human translation has gained interest in both the MT and professional translation communities. The availability of reliable MT that can serve as a starting point for human translation can potentially save countless hours for both professional and volunteer translators worldwide. Venues such as the 2012 AMTA Workshop on Post-Editing Technology and Practice (O'Brien et al., 2012) showcase a variety of innovative approaches to tighter integration of MT with human translation workflows and analyses of the impact of MT on professional translation.

Professional translation projects typically follow the three stage process of translation, editing, and proofreading to ensure high quality results. Most approaches to computer-aided translation (CAT) target the first stage, using real-time MT systems to provide sentence-level translations for humans to post-edit. This section describes initial work that examines the impact on human translation when output from existing MT systems is provided to human translators. Several lines of related work are covered in other chapters: work on systems that learn from translator feedback is discussed in §3.1, work on predicting the usefulness of translations for human editing is discussed in §4.1, and work on CAT tools is discussed in §5.1.

1.4.1 Machine Translation Post-Editing in Human Workflows

Human translators typically employ translation memory (TM) systems that archive previously translated sentences in the same domain. TM systems use "fuzzy matching" algorithms based on edit distance to locate translations of sentences similar to the source sentence being translated. If a sufficiently similar sentence is found, the translation is suggested. While the translation may not be accurate for the current sentence, it is guaranteed to be a human-quality translation of something. Conversely, MT systems always produce suggestions for every sentence, but have no quality guarantees. He et al. (2010) conduct a user study that integrates machine translation into a TM system used by human translators. For each sentence, translators are presented with both a human translation from the TM and a MT hypothesis and asked to select the translation that was most suitable for post-editing. The authors find that both TM and MT outputs are selected regularly and in some cases, translators are unable to tell the difference. This result shows promise for the idea of using MT systems to improve TM coverage.

Several organizations have conducted evaluations on the productivity benefits of adding MT to their translation editing workflows. Zhechev (2012) describes experiments comparing post-editing to translating from scratch when localizing Autodesk[3] software. Post-editing significantly improves translation efficiency in several language directions. Poulis and Kolovratnik (2012) describe experiments for the European Parliament that add machine translation to existing tools such as translation memories and bilingual dictionaries.

[2] http://en.wikipedia.org/wiki/Wikipedia:Translate_us
[3] http://www.autodesk.com/

Results are mixed, with MT yielding improved results for some language directions. Tatsumi (2010) conducts a large scale study of MT post-editing in real-world scenarios with professional Japanese translators. Results show that suggestions from a MT system tend to require the same amount of post-editing as "good" matches (above 75% similarity) from a TM when translating in the information technology domain. For TM matches, editors mostly edit lexical items while for MT output, editors mostly fix grammatical errors. Finally, Tatsumi et al. (2012) examine the effectiveness of "crowd-sourcing" post-editing (employing non-experts over the Internet). The authors find that when given MT output, larger pools of non-experts can frequently produce "good enough" translations at least as quickly as experts, often for little or no cost to community projects such as localizing websites.

The general consensus of post-editing studies is favorable: while there is still much room for improvement, the introduction of machine translation tends to improve human translation productivity. Even in scenarios where assistive technologies such as bilingual dictionaries and translation memories are already present, the addition of MT leads to quantifiable gains.

1.4.2 Analysis of Post-Editing

In an application popularized by the DARPA Global Autonomous Language Exploitation (GALE) project (Olive et al., 2011), translation post-editing can been used as a semi-automatic evaluation metric. In human-targeted translation edit rate (HTER) (Snover et al., 2006), a human minimally edits a translation hypothesis such that it is grammatical and meaning-equivalent with a reference translation. The TER metric is then used to approximate the number of atomic operations required to post-edit the sentence. In this case, the human editor does not need to be bilingual. In the context of evaluation, HTER is cast as the *minimum distance* between a system's output and any possible translation. However, HTER has also been used as an approximation for the amount of human effort required for post-editing in tasks such as the 2012 NAACL Workshop on Statistical Machine Translation Quality Estimation task (Callison-Burch et al., 2012).

Other work focuses directly on quantifying post-editing effort. Koponen et al. (2012) show that longer post-editing times are correlated with types of errors that translators rate as more difficult to edit. Lacruz et al. (2012) connect longer pauses in post-editing activity with more challenging edits; translators need to spend more time mentally processing difficult cases before mechanically editing the translation. Blain et al. (2011) take a more qualitative approach to understand post-editing by introducing a measure based on *post-editing actions*. Edits are grouped into linguistically interpretable actions such as NP structure change, verb agreement correction, and multi-word expression correction.

Chapter 2

Introduction

2.1 Thesis Statement

We have introduced the components of current state-of-the-art statistical machine translation systems and discussed initial efforts to integrate MT with human translation workflows, specifically by providing initial translations for humans to edit. While general improvements in MT quality have led to increased interest in this application, there has been little work on designing translation systems specifically for post-editing. In this work, we propose improvements to key components of MT pipelines aimed at significantly reducing the amount of work required from human translators. We claim that:

- the amount of work required of human translators can be reduced by translation models that immediately learn from editor feedback,
- the amount of work required of human translators can be reduced by identifying the most costly types of translation errors and tuning MT systems to avoid them,
- and that the amount of work required of human translators can be better *quantified* with more accurate statistical measures.

To support these claims, we propose to:

- develop an online translation system that immediately incorporates post-editor feedback,
- develop an extended translation feature set that allows the model to evaluate different sources of feedback over time,
- develop advanced automatic metrics capable of predicting post-editing effort for MT system optimization and evaluation,
- conduct an in-depth analysis of various types of post-editing measures and measure combinations to determine the most reliable method for evaluating human editing effort,
- and directly investigate the impact of online learning on post-editing requirements in a real-time translation scenario with human translators.

The completion of this work will benefit both human and machine translation communities. Our online model adaptation techniques and automatic metrics can be used with other translation systems and generalized to other work that aims to improve MT for post-editing. Our data collection and analysis will provide valuable insight into the mechanics of human post-editing. Finally, our end-to-end translation system optimized for post-editing can be plugged in as a resource to computer-aided translation environments for use by translators around the world.

2.2 Experimental Framework

To evaluate the impact of our work, we have assembled a test suite that represents a variety of real-world translation scenarios in four language directions: from English into and out of Spanish and Arabic. For each direction, we have extensive bilingual and monolingual data for model estimation as well as in-domain and out-of-domain evaluation sets with reference translations that can be used to simulate post-editing. We also build a traditional machine translation system for each scenario to serve as both a baseline to compare results against and as a platform for implementing our extensions to standard translation models. Data and tools are selected with a focus on reproducibility. Spanish–English resources are freely available online and Arabic–English resources are available with a Linguistic Data Consortium (LDC) subscription. All tools other than MADA are freely available under open source licenses.

Tools: We use several natural language processing toolkits to process training data and build translation systems. Part of our work includes significant contributions to some of these tools, particularly the suffix array grammar extractor, Meteor, and TransCenter.

- `cdec`: a statistical machine translation framework including a SCFG decoder, a word aligner, and implementations of several learning algorithms for structured prediction models (Dyer et al., 2010; Dyer et al., 2013). `cdec` is released under the Apache License at `http://github.com/redpony/cdec`.

- KenLM: a toolkit for estimating and conducting inference with N-gram language models (Heafield, 2011). KenLM is released under the GNU LGPL at `kheafield.com/code/kenlm/` and also included with `cdec`.

- MADA: an Arabic natural language processing toolkit for tokenization, diacrization, morphological disambiguation, part-of-speech tagging, stemming and lemmatization (Habash et al., 2009). MADA is released under a non-commercial use license at `http://www1.ccls.columbia.edu/MADA/`.

- Meteor: an automatic metric for evaluation and optimization of machine translation systems (Banerjee and Lavie, 2005; Denkowski and Lavie, 2011). Meteor is released under the GNU LGPL at `http://github.com/mjdenkowski/meteor`.

- MultEval: implementation of several statistical significance tests for machine translation evaluation (Clark et al., 2011). MultEval is released under the GNU LGPL at `http://github.com/jhclark/multeval`.

- Suffix array grammar extractor: an implementation of suffix array-based grammar extraction for hierarchical phrase-based machine translation that has been repackaged as part of `cdec` (Lopez, 2008a; Chahuneau et al., 2012).

- TransCenter: a web-based framework for collecting and analyzing human post-editing information (Denkowski and Lavie, 2012b). TransCenter is released under the GNU LGPL at `http://github.com/mjdenkowski/transcenter`.

Data: We select four language directions for translation post-editing experiments: Spanish-to-English, English-to-Spanish, Arabic-to-English, and English-to-Arabic. Bilingual resources are identical between directions for each language pair while monolingual resources are unique for each language direction. Training data for Spanish–English includes all constrained resources for the 2012 NAACL Workshop on

	Training Data		Evaluation Sets (sents)			
	Bilingual (sents)	Monolingual (words)	*WMT10*	WMT11	TED1	TED2
Spanish–English	2,104,313	1,175,142,205	*2489*	3003	2688	2978
English–Spanish	2,104,313	304,262,351	*2489*	3003	2688	2978
	Training Data		Evaluation Sets (sents)			
	Bilingual (sents)	Monolingual (words)	*MT08*	MT09	TED1	TED2
Arabic–English	5,027,793	651,957,491	*1356*	1313	2690	2846
English–Arabic	5,027,793	168,323,504	*1356*	1313	2690	2846

Table 2.1: Training, development, and evaluation data sizes for all experimental scenarios. Italics indicate that a data set is used for system optimization. The MT08 and MT09 sets have 4 English reference translations for each Arabic source sentence. All other data sets, including the English–Arabic directions of MT08 and MT09, have a single reference for each source sentence.

Statistical Machine Translation (WMT)[1] (Callison-Burch et al., 2012), consisting of European parliamentary proceedings and news commentary. Training data for Arabic–English includes all constrained bilingual resources for the 2012 NIST Open Machine Translation Evaluation (OpenMT12)[2] (Przybocki, 2012), consisting largely of news, and a selection from the English Gigaword (Parker et al., 2011). For each language direction, we have four evaluation sets: two drawn from similar domains as the training data, and two drawn from broader domains. For similar-domain sets, we use the 2010 and 2011 WMT news test sets for Spanish–English and the 2008 and 2009 OpenMT mixed news and weblog test sets for Arabic–English. For broad-domain sets, we use sections of the Web Inventory of Transcribed and Translated Talks (WIT[3]) corpus[3] (Cettolo et al., 2012) that contains multilingual transcriptions of TED[4] talks. The test sets *TED1* and *TED2* each contain bilingual transcriptions of 10 TED talks delivered from a wide range of speakers on a variety of topics. All data sets are pre-segmented into sentences that are grouped by document. We apply further *tokenization*, splitting sentences into individual words that can be processed by alignment and translation models. English and Spanish are tokenized with a general-purpose tokenizer included with *cdec* while Arabic is tokenized using MADA. Details for all training and evaluation data sets after tokenization are shown in Table 2.1.

System Building: Translation systems are built using the same methods for each language pair. Data is word aligned source-to-target and target-to-source using the `fast_align` word aligner included with `cdec` (Dyer et al., 2013). Alignments are symmetrized using the `grow-diag-final-and` heuristic (Axelrod et al., 2005). Translation grammars with the standard feature set listed in Table 2.2 (see §1.2.2–1.2.4 for feature descriptions) are extracted using the suffix array grammar extractor. Unpruned, modified Kneser-Ney smoothed (Chen and Goodman, 1996) language models are estimated using KenLM. Feature weights are learned using the implementation of lattice-based minimum error rate training (Och, 2003; Macherey et al., 2008) included with `cdec`. MERT optimizes the BLEU score (Papineni et al., 2002) on the development set for the language pair (WMT10 for Spanish–English or MT08 for Arabic–English).

Automatic Evaluation: We use the `cdec` decoder to translate all evaluation sets into the target language using the same set of feature weights learned from the development set. To simulate a scenario where no in-domain data is available, we do not re-tune systems for TED talks. Translations are evaluated automatically

[1] http://statmt.org/wmt12/translation-task.html
[2] http://www.nist.gov/itl/iad/mig/openmt12.cfm
[3] https://wit3.fbk.eu/
[4] http://www.ted.com/pages/about

	Feature	Definition
Phrase Features	`CoherentP(e|f)`	Equation 1.14
	`Count(f,e)`	Equation 1.11
	`SampleCount(f)`	Equation 1.12
	`Singleton(f)`	Equation 1.13
	`Singleton(f,e)`	Equation 1.13
Lexical Features	`MaxLex(e|f)`	Equation 1.10
	`MaxLex(f|e)`	Equation 1.10
Language Model Features	`LM(E)`	Equation 1.15
	`LM_OOV(E)`	Equation 1.16
Derivation Features	`Arity(0)`	§ 1.2.4
	`Arity(1)`	§ 1.2.4
	`Arity(2)`	§ 1.2.4
	`GlueCount`	§ 1.2.4
	`PassThroughCount`	§ 1.2.4
	`WordCount`	§ 1.2.4

Table 2.2: Standard (baseline) feature set for hierarchical phrase-based machine translation with suffix array grammars

with BLEU (Papineni et al., 2002), TER (Snover et al., 2006), and Meteor 1.4 (Denkowski and Lavie, 2011). To account for optimizer instability, we run MERT 3 times for each language direction and decode the evaluation sets with each set of weights. The reported metric score for a data set is the average of three tune-test cycles. When comparing extended systems to baselines, we use the techniques described by Clark et al. (2011) to test for statistically significant differences in score.

Human Evaluation: In addition to automatic evaluations of translation quality, we will conduct real-time human post-editing experiments comparing our extended systems to baselines. For at minimum one language direction (more scenarios as resources allow), human translators will edit output from both a non-adaptive baseline system and our best performing extended system. Metrics such as post-editing time, number of keystrokes, and average pause ratio will be used to compare the amount of effort required to edit each system's output. Data collection and analysis is further described in Chapter 5.

Chapter 3

Online Learning for Machine Translation

When a machine translation system outputs hypotheses for human post-editing, every translation error costs time to correct. Ideally, a translation system should be able to learn from correction and avoid making the same mistakes repeatedly. Every time a human translator corrects a machine translation hypothesis, a new bilingual sentence pair is produced. However, the standard translation models described in Chapter 1 are not designed to incorporate this feedback. These models use the *batch* learning paradigm, wherein all training data is used to estimate a model (translation system) and the model is then use to make predictions (translation of new input sentences). Incorporating new data into the model requires pooling it with initial training data, rerunning word alignment, and estimating new translation models. As this process is computationally expensive, it is typically weeks, months, or longer between system re-trainings. This leads to post-editors spending their time correcting the same translation errors repeatedly.

Alternatively, the task of machine translation for post-editing can be cast as an *online* learning task. In the online learning paradigm, a task proceeds as a series of trials. Each trial consists of the following three stages: (1) the model makes a prediction, (2) the model receives the "true" answer, and (3) the model updates its parameters. Post-editing workflows fit naturally into this paradigm. In the prediction stage, the translation system produces an initial hypothesis. A human post-editor then edits the hypothesis to produce the "true" translation. Finally, the system uses the new source-target sentence pair to update the translation model. While the process of post-editing naturally produces the bilingual sentence pairs in step 2, traditional translation models are not equipped to incorporate this data. In this chapter, we describe an online translation model that immediately incorporates new training instances, allowing it to learn from human feedback and avoid making the same mistakes repeatedly.

3.1 Related Work

Initial work has led to the successful extension of standard phrase-based systems to immediately incorporate post-edit feedback. Ortiz-Martínez et al. (2010) describe a method for handling new bilingual sentence pairs generated during translation. In addition to feature scores, sufficient statistics are kept for phrase pairs in the translation model. When a new sentence pair is available, it is aligned with an iterative word alignment model and new phrase instances are extracted. These instances are used to add new phrase pairs to the model and update the appropriate sufficient statistics that are in turn used to recompute feature scores. López-Salcedo et al. (2012) introduce a discriminative learning method for updating a phrase-based system's feature weights with feature scores held constant. Weights are adjusted to favor hypotheses that are closer to human post-edited translations. Martínez-Gómez et al. (2012) examine adapting both feature functions and feature weights. Sanchis-Trilles (2012) proposes a strategy for online language model adaptation wherein several smaller domain-specific models are built and their scores interpolated for each sentence translated.

Feature	On-Demand	Online
`CoherentP(e|f)`	$\log\left(\frac{\mathcal{C}_{\mathcal{S}}(\bar{f},\bar{e})}{\lvert\mathcal{S}\rvert}\right)$	$\log\left(\frac{\mathcal{C}_{\mathcal{S}}(\bar{f},\bar{e})+\mathcal{C}_{\mathcal{L}}(\bar{f},\bar{e})}{\lvert\mathcal{S}\rvert+\lvert\mathcal{L}\rvert}\right)$
`SampleCount(f)`	$\log\left(\lvert\mathcal{S}\rvert\right)$	$\log\left(\lvert\mathcal{S}\rvert+\lvert\mathcal{L}\rvert\right)$
`Count(f,e)`	$\log\left(\mathcal{C}_{\mathcal{S}}(\bar{f},\bar{e})\right)$	$\log\left(\mathcal{C}_{\mathcal{S}}(\bar{f},\bar{e})+\mathcal{C}_{\mathcal{L}}(\bar{f},\bar{e})\right)$
`Singleton(f)`	$\mathcal{C}_{\mathcal{S}}(\bar{f})=1$	$\mathcal{C}_{\mathcal{S}}(\bar{f})+\mathcal{C}_{\mathcal{L}}(\bar{f})=1$
`Singleton(f,e)`	$\mathcal{C}_{\mathcal{S}}(\bar{f},\bar{e})=1$	$\mathcal{C}_{\mathcal{S}}(\bar{f},\bar{e})\ +\ \mathcal{C}_{\mathcal{L}}(\bar{f},\bar{e})=1$
`PostEditSupport(f,e)`	0	$\mathcal{C}_{\mathcal{L}}(\bar{f},\bar{e})>0$

Table 3.1: Phrase feature definitions for on-demand and online translation models.

Interpolation weights depend on the domain of the sentence being translated, allowing the decoder to trust more relevant monolingual data for each sentence.

Hardt and Elming (2010) demonstrate the effectiveness of maintaining a distinction between background data (the initial data used to build systems) and post-edit data in an online system. The authors also show the feasibility of using pre-existing human reference translations to conduct simulated post-editing experiments, proceeding as follows. A system with both a standard and post-edit-specific phrase table translates sentences in a data set. After each sentence is translated, it is aligned to a reference translation as a substitute for post-editing and phrases extracted from the new sentence pair are added to the post-edit-specific phrase table similarly to Ortiz-Martínez et al. (2010).

While not expressly targeting the application of post-editing, Levenberg et al. (2010) describe a method for incorporating new bilingual data into on-demand grammar extraction as it is available. Introducing a version of the suffix array data structure that can be dynamically updated, the authors are able to add to the data pool from which sentence-level translation grammars are sampled.

3.2 Completed Work: Online Translation Model Adaptation

In this section, we introduce an online version of a rich hierarchical phrase-based translation model that immediately incorporates human feedback by learning new translation rules from post-edited output. This model is a straightforward extension of the suffix array-backed on-demand model described in §1.2.3 (Lopez, 2008a). Our model offers three key advantages over previous work on online translation models. First, by moving from a traditional phrase-based model to a hierarchical model, rules learned from post-edited data can encode non-local *reordering* in addition to new translation choices. Second, our model maintains all sufficient statistics required to use the powerful suffix array-backed features described in §1.2.3. These features are shown by Lopez (2008b) to significantly outperform the standard feature set used in prior models. Third, by using existing bilingual text to *simulated* post-editing, we can learn appropriate weights for our online feature set using standard optimization algorithms such as minimum error rate training.

3.2.1 Grammar Extraction

The starting point for our model is the on-demand translation model described in §1.2.3 (Lopez, 2008a; Lopez, 2008b). Rather than using all bilingual training data to build a single, large translation grammar, this approach uses a suffix array to index the data so that grammars can be estimated as needed. When a new sentence needs to be translated, the suffix array is used to rapidly build and score a sentence-specific grammar. Rules in on-demand grammars are generated using a sample $\mathcal{S}$ for each source phrase $\bar{f}$ in the input sentence. The sample, containing phrase pairs $\langle \bar{f}, \bar{e} \rangle$, is used to calculate the following statistics:

- $\mathcal{C}_{\mathcal{S}}(\bar{f}, \bar{e})$: count of instances in $\mathcal{S}$ where $\bar{f}$ aligns to $\bar{e}$ (phrase co-occurrence count).
- $\mathcal{C}_{\mathcal{S}}(\bar{f})$: count of instances in $\mathcal{S}$ where $\bar{f}$ aligns to any target phrase.
- $|\mathcal{S}|$: total number of instances in $\mathcal{S}$, equal to number of occurrences of $\bar{f}$ in training data, up to the sample size limit.

These statistics are used to instantiate translation rules $X \rightarrow \bar{f} / \bar{e}$ and calculate scores for the phrase feature set shown in the "on-demand" column of Table 3.1, including the powerful *coherent* translation score. Although grammars are not sampled until needed, the suffix array is pre-indexed and does not facilitate adding new data.

To accommodate new bilingual data from post-editing, our system maintains a dynamic lookup table in addition to the static suffix array. When a human translator edits a translation hypothesis, the sentence pair resulting from the input sentence and post-edited translation is word-aligned with the same model used for the initial data. This process (often called forced alignment) is the only approximation in our model with respect to the on-demand model. As statistical word alignment is an unsupervised task, the existence of an additional sentence pair in the data can impact its alignment. However, with sufficiently large initial data, forced alignments are sufficiently accurate for learning translation models. Aligned phrase pairs are stored in the lookup table and phrase occurrences are counted on the source side. When a new grammar is extracted, our model uses all training instances extracted from previously post-edited sentences to learn translation rules. The suffix array sample $\mathcal{S}$ for each $\bar{f}$ is accompanied by an exhaustive lookup $\mathcal{L}$ from the lookup table. Statistics matching those from $\mathcal{S}$ are calculated from $\mathcal{L}$:

- $\mathcal{C}_{\mathcal{L}}(\bar{f}, \bar{e})$: count of instances in $\mathcal{L}$ where $\bar{f}$ aligns to $\bar{e}$.
- $\mathcal{C}_{\mathcal{L}}(\bar{f})$: count of instances in $\mathcal{L}$ where $\bar{f}$ aligns to any target phrase.
- $|\mathcal{L}|$: total number of instances of f in post-editing data (no size limit).

Combined statistics from $\mathcal{S}$ and $\mathcal{L}$ are used to calculate the "online" feature set defined in Table 3.1. An additional indicator feature `PostEditSupport(f,e)` marks rules that are consistent with post-editor feedback. As this model is hierarchical, phrase pairs in $\mathcal{L}$ can contain other phrase pairs, allowing the model to learn new non-local language phenomena from post-editor feedback.

Like work by Levenberg et al. (2010), this learning process can be seen as influencing the *distribution* from which on-demand grammars are sampled over time. The resulting translation grammars are identical (subject to word alignments and sampling strategy) to the infeasible process of adding each instance to the initial training data, re-aligning, and recompiling the suffix array after every sentence is translated. Our model has the added advantage of tracking which data comes from post-edited data. As this data is likely to be highly relevant to subsequent input sentences, we require all relevant phrase pairs from the lookup table to be included when sampling new grammars. This can be seen as *biasing* the statistical model to prefer data that is more likely to yield relevant translation rules. Additionally, the `PostEditSupport(f,e)` feature allows an optimizer to learn an additional weight for all rules that are consistent with human feedback.

	Spanish–English				English–Spanish			
	WMT10	WMT11	TED1	TED2	*WMT10*	WMT11	TED1	TED2
On-Demand	*29.2*	27.9	32.8	29.6	*27.4*	29.1	26.1	25.6
Online	***30.2***	**28.8**	**34.8**	**31.0**	***28.5***	**30.1**	**27.8**	**27.0**
	Arabic–English				English–Arabic			
	MT08	MT09	TED1	TED2	*MT08*	MT09	TED1	TED2
On-Demand	*38.0*	41.6	10.5	10.5	*18.9*	23.8	7.5	7.9
Online	***38.5***	**42.3**	**11.3**	**11.7**	***19.2***	**24.1**	**8.0**	**8.7**

Table 3.2: BLEU scores for baseline and online translation systems on news and TED data in a variety of language scenarios. Reported scores are averages over three MERT runs. Bold numbers indicate statistically significant improvement. Italics indicate tuning data.

In addition to marking new rules, this feature facilitates *disambiguation* of existing rules. For example, if a source phrase has 10 possible translations with similar scores and only one is acceptable in the current context, the model will struggle to select the correct choice. However, if that choice is marked as supported by editor feedback and the model has learned to trust this feature, it has a greater chance of producing a correct translation.

3.2.2 Parameter Optimization

Optimization strategies such as minimum error rate training (Och, 2003) are used to learn optimal feature weights for statistical translation models. However, these algorithms require a tuning set to be translated repeatedly with different feature weights, making parameter optimization for human-dependent post-editing systems difficult or impossible. Following Hardt and Elming (2010), we formulate the task of simulated post-editing wherein pre-generated human reference translations are used as a stand-in for actual post-edited translations. This allows any translation hypothesis to be instantly "post-edited" by copying over the reference translation. Prior to optimization, one set of grammars is extracted, one per sentence in the tuning set. After each grammar is extracted, the source sentence is aligned to the reference translation, forming a simulated post-editing data point, and incorporated into the model. As all post-edit information is pre-encoded in the grammars and reference translations, MERT, or any comparable optimizer can be run normally, learning a weight for each feature. Despite the fact that learning a single weight for a feature that becomes more powerful as more sentences are translated limits the effectiveness of standard linear translation models, our initial experiments show that MERT is capable of finding weights that lead to significant improvement in translation quality. These weights can be seen as an average of a feature's effectiveness over time. For the first sentence of a document, grammar rules and feature scores are identical to the on-demand model's. As more sentences are translated, new rules are learned and feature scores are more accurate for the current context.

3.2.3 Results

We evaluate our online translation model in all scenarios outlined in §2.2, translating a mixture of language directions and domains that cover a broad range of difficulty levels. Shown in Table 3.2, our online translation model yields significant improvement over an on-demand baseline in all experiments. Gains are larger for TED talks where translator feedback can bridge the gap between domains. Table 3.3 shows the aggregate percentages of rules in online grammars that are entirely new (extracted from post-edit instances only) or post-edit supported (superset of new rules). While percentages vary by language and data set, the

	News		TED Talks	
	New	Supp	New	Supp
Spanish–English	15%	19%	14%	18%
English–Spanish	12%	16%	9%	13%
Arabic–English	9%	12%	23%	28%
English–Arabic	5%	8%	17%	20%

Table 3.3: Percentages of new rules and post-edit supported rules (both old and new rules for which the `PostEditSupport(f,e)` feature fires) in online grammars by domain.

overall trend is a combination of *learning* new vocabulary and reordering and *disambiguating* existing translation choices. On average, online grammar extraction requires 10% additional CPU time per grammar and negligible memory, keeping real-time learning and translation viable for live post-editing scenarios.

3.3 Proposed Work: Rich Feature Sets for Model Adaptation

While our online translation model shows good improvement over an on-demand baseline, there are two obvious shortcomings. First, the model only distinguishes between initial and post-edited data. A single lookup table maintains data for all translations across documents and domains. Second, the model requires a single weight per feature despite features' becoming more powerful over time. This prevents the model from placing more trust in feature scores as they become more accurate. In our proposed work, we will address these issues by introducing a more expressive, expanded feature set for the same underlying translation model.

3.3.1 Domain-Specific Post-Edit Features

While our current model treats input sentences as independent samples, natural language text is frequently organized into documents. Within documents, sentences are more likely to share vocabulary, context, and grammatical style. In the WMT, OpenMT, and TED talk evaluation sets, documents are clearly defined. Rather than treating an entire input set as a single stream, we will extend our model to consider different translation contexts (referred to as domains). Daume III (2007) introduces a technique for domain adaptation via multiplied feature sets. This technique can be applied when data C (for translation, bilingual text) is taken from multiple domains. In our case, we will divide data into three groups: initial data C^I indexed in the suffix array, post-edit data from the current translator for the current document C^D, and post-edit data from the current translator for all documents C^T. In the base case, this data is aggregated and used to score a single feature set. Following Daume III, we will also calculate multiple copies of our feature set, each scored on only data from a certain domain. Creating multiple copies of each feature allows an optimizer to learn to weigh information from different data sources independently. In addition to simply biasing the model toward post-edit data by changing the sampling strategy, we will use simulated post-editing to directly learn weights for initial, translator, and document data that optimize translation performance. Similar domain adaptation techniques have been shown to work well in machine translation by Clark et al. (2012).

We accommodate this domain adaptation by generalizing the phrase feature set to include an arbitrary number of data sources. For each type of post-edit data, we maintain a separate lookup table that is updated as new matching training instances arrive (for example, sentence pairs for a current document are recorded in tables for both C^D and C^T). When learning new grammars, we use generalized data selections $\mathcal{S}_j$ (either samples or exhaustive lookups) from each relevant data source in a set J to score an instance of the

Feature	Scoring Function
`CoherentP(e|f)_J`	$\log\left(\frac{\sum_j \mathcal{C}_{\mathcal{S}j}(\bar{f},\bar{e})}{\sum_j \lvert\mathcal{S}j\rvert}\right)$
`SampleCount(f)_J`	$\log\left(\sum_j \lvert\mathcal{S}j\rvert\right)$
`Count(f,e)_J`	$\log\left(\sum_j \mathcal{C}_{\mathcal{S}j}(\bar{f},\bar{e})\right)$
`Singleton(f)_J`	$\sum_j \mathcal{C}_{\mathcal{S}j}(\bar{f}) = 1$
`Singleton(f,e)_J`	$\sum_j \mathcal{C}_{\mathcal{S}j}(\bar{f},\bar{e}) = 1$
`DataSupport(f,e)_J`	$\sum_j \mathcal{C}_{\mathcal{S}j}(\bar{f},\bar{e}) > 0$

Table 3.4: Phrase feature definitions for generalized multiple data source grammars.

generalized feature set defined in Table 3.4. This extension facilitates not only our own work, but any work using an on-demand model that draws from multiple data sources.

3.3.2 Data Size Post-Edit Features

As more sentences are post-edited, the amount of data for a given translator and document grows, leading to more reliable domain-specific statistics. Currently, an optimizer must learn a single weight for each source, forcing an averaging effect over time. To allow time-specific features, we will make multiple copies of each *domain specific* feature set, labeled by number of training instances collected. For example, we can copy a feature set $\mathcal{H}$ for cases where the associated data source contains 0 to j, j to k, and k or more post-edited sentence pairs. When a new grammar is estimated, the number of training instances i in the associated source is used to determine which copy of $\mathcal{H}$ fires. Other copies of the feature set return 0. Formally, the score for the copy of feature set $\mathcal{H}$ covering data size j through k at current size i is given:

$$\mathcal{H}_j^k(X \rightarrow \bar{f}/\bar{e}, i) = \begin{cases} \mathcal{H}(X \rightarrow \bar{f}/\bar{e}) & \text{if } j \leq i \leq k \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Under this configuration, only one set of features fires for a given source at a given time, allowing an optimizer to increase the weight of a data source as it becomes more informative. We will experiment with multiple numerical ranges to determine good time splits for post-editing data sources.

3.3.3 Evaluation

To evaluate the impact of expanded feature sets, we will conduct translation experiments in the scenarios described in §2.2. As the size of our feature sets now exceeds what minimum error rate training can reliably optimize, we move to the pairwise rank optimization algorithm for learning feature weights. To distinguish between changes in optimizer and changes in feature set, we will compare to a version of our simple online system tuned using PRO. This entails running the following tune-test cycles for all scenarios:

- Basic feature set tuned with PRO
- Extended feature sets by source: initial, document, translator, general
- Extended feature sets by source and size (ideal ranges to be determined)

As some initial experiments with PRO have yielded mixed results, we will also test the large margin learning algorithm described by Eidelman (2012), which also scales to large feature sets and has been reported to be more stable. The best performing system from the listed experiments will also be used for real-time post-editing experiments described in §5.4 to determine impact on actual human translation.

Chapter 4

Optimizing Machine Translation for Post-Editing

Traditionally, machine translation is treated as a final product that humans will use to read content in their native languages and other language technologies such as information retrieval systems will use directly as input. Approaches to both human and automatic evaluation focus on improving the adequacy of MT system output for these purposes. In contrast, post-editing uses MT as an intermediate step to reduce the amount of work required by human translators. Whereas translation models that incorporate post-edit feedback target this task in terms of model estimation, automatic metrics that accurately evaluate the amount of work required to edit translation hypotheses target post-editing in terms of parameter optimization. Pairing online models with automatic post-editing metrics enables end-to-end translation systems specifically targeting human translation. In this chapter, we discuss the differences between traditional adequacy-driven MT tasks and post-editing, highlighting the need for improved optimization and evaluation targets. We then introduce our extended version of the Meteor metric, shown to correlate well with human post-editing assessments. We also report results for tuning MT systems using Meteor, showing improved stability over standard tuning metrics in minimum error rate training. We describe planned experiments for collecting more detailed assessments of human post-editing effort and using this data to tune new versions of Meteor. These versions of Meteor will also be used as objective functions to tune our online systems with extended feature sets.

4.1 Related Work

4.1.1 Evaluation

As discussed in §1.3.3, metrics that automatically assign quality scores to translation hypotheses are vital in both parameter optimization and system evaluation. While standard metrics are engineered to correlate with human judgments of translation adequacy, recent work has aimed to predict the amount of post-editing required for MT output, both with and without pre-generated reference translations. Most work measures editing effort with human-targeted translation edit rate (§1.4.2) (Snover et al., 2006). The largest scale evaluation of reference-based evaluation metrics' ability to predict HTER is the 2010 ACL Joint Workshop on Statistical Machine Translation and MetricsMATR (Callison-Burch et al., 2010), in which several metrics are shown to outperform standard BLEU in predicting HTER at the sentence level. The metric with the highest correlation is TER-plus (Snover et al., 2009), an extension of TER that uses word stemmers, synonym dictionaries, and probabilistic paraphrase tables to add various types of weighted substitution operations

to edit distance calculation. The Stanford probabilistic edit distance evaluation metric[1] (Wang and Manning, 2012), another weighted edit distance metric using flexible linguistic features including synonymy and paraphrasing, also performs well above the BLEU baseline.

While reference-based evaluation metrics are ideal for system optimization, *quality estimation* metrics that only consider a source sentence and translation hypothesis are useful for deciding if a given translation from a MT system is useful for post-editing. Specia and Farzindar (2010) predict sentence-level HTER using support vector machines with a variety of features including source and target length, N-gram language model scores, number of translations per word in probabilistic dictionaries, and mismatches in punctuation. The authors report good correlation with HTER for translations between English, French, and Spanish. In later work, Specia (2010) combines quality estimation features with several well-known reference-based metrics to train a classifier that predicts sentence-level HTER with improved accuracy. The 2012 NAACL Workshop on Statistical Machine Translation (Callison-Burch et al., 2012) features a quality estimation task where participants predict the amount of post-editing required for sentence-level translations of English into Spanish as assessed by professional translators. No reference translations are provided. Well-performing entries employ a variety of features including measuring translation similarity to outputs of other MT systems (Soricut et al., 2012) and measuring similarity between constituency and dependency parses of source and target sentences (Hardmeier et al., 2012).

As post-editing effort is widely measured by either HTER or professional translators' assessments of editing difficulty, recent work has examined how reliable these measures are for training and evaluating natural language processing systems. Specia and Gimenez (2010) and Koponen (2012) observe cases where human assessments differ from mechanically calculated HTER scores. In cases where sentences are very long or where significant reordering is required, translators tend to classify a sentence as much more difficult to post-edit than its HTER score would indicate. Koponen hypothesizes that these cases require increased cognitive effort that is not adequately captured by simple edit distance. Specia (2011) compare the suitability of post-editing time, distance (HTER), and score (human assessment) for training predictive models. Effort and time are both shown to outperform edit distance, although differences in data sets between languages make generalization difficult.

4.1.2 Optimization

As work on automatic metrics has largely focused on evaluation and quality estimation, the task of using new metrics as objective functions for optimization algorithms has been less explored. Initial work by Cer et al. (2010) shows that optimizing a MT system to a given evaluation metric generally increases translation score on that metric, often at the expense of scores on other metrics. The work shows that there is no compelling reason to move away from standard BLEU at the time. Liu et al. (2011) report improvement when applying the TESLA (translation evaluation of sentences with linear-programming-based analysis) metric to a MERT task. Based on N-gram matches like BLEU, the TESLA metric also allows flexible matching with synonyms and part-of-speech tags and weights N-gram matches based on match type and presence of content words. In a human evaluation, annotators prefer translations from a TESLA-tuned system over a BLEU-tuned system despite the fact that these translations generally receive lower BLEU scores. The 2011 EMNLP Workshop on Statistical Machine Translation (Callison-Burch et al., 2011) features a system optimization task where participants use a provided MERT implementation and development set to tune an existing translation system to various automatic metrics. By human evaluation, no metric outperforms standard BLEU, though several perform comparably.

[1]This metric is described as "Stanford" in the official results of WMT/MetricsMATR 2010. The first published description by the authors is in 2012 under the name "SPEDE".

4.2 Completed Work: Meteor Metric for Evaluation and Optimization

Originally developed to more accurately model human acceptability of MT output, the Meteor metric (Banerjee and Lavie, 2005; Lavie and Denkowski, 2009) has consistently shown good correlation with human judgments of adequacy (Lavie and Agarwal, 2007) and preference (Agarwal and Lavie, 2008; Denkowski and Lavie, 2010b; Denkowski and Lavie, 2011). This section describes our extended version of Meteor, casting its features as predictors of post-editing effort. We report results for successfully adapting Meteor to predict HTER scores while at the same time revealing limitations of HTER as a measure of editing effort. We also discuss initial system tuning experiments that show a stability advantage over BLEU.

4.2.1 The Meteor Metric

Meteor is an automatic evaluation metric that scores MT hypotheses by aligning them to reference translations. Alignments are based on several types of flexible matches that go beyond surface forms to identify word and phrase correspondences that would be clear to humans but are missed by standard metrics. Based on an alignment, Meteor determines what information is present in both sentences, what information is present in one sentence but not the other, and to what extent text is reordered between the two sentences. Alignment statistics are combined in a weighted scoring function that can be tuned to maximize correlation with human judgments of translation quality.

Meteor Alignment: Given a translation hypotheses E' and reference translation E, Meteor creates an alignment as follows. First, the search space of possible alignments is constructed by identifying all possible matches between the two sentences according to the following matchers:

- Exact: Match words if their surface forms are identical.
- Stem: Stem words using a language-appropriate Snowball stemmer (Porter, 2001) and match if the stems are identical.
- Synonym: Match words if they share membership in any synonym set according to the WordNet (Miller and Fellbaum, 2007) database.
- Paraphrase: Match phrases if they are listed as paraphrases in the Meteor paraphrase tables. Paraphrase tables are constructed from the bilingual text available as part of the 2010 ACL Workshop on Statistical Translation and MetricsMATR (Callison-Burch et al., 2010) using the statistical phrase table "pivot" approach (Bannard and Callison-Burch, 2005) with additional pruning to improve precision (Denkowski and Lavie, 2011).

All matches are generalized to phrase matches in the form $\langle {E'}_i^{i+n}, E_j^{j+m} \rangle$ where i and j are start indices in the hypothesis and reference and n and m are match lengths. Matches are said to *cover* one or more words in each sentence. Exact, stem, and synonym matches always cover one word in each sentence while paraphrase matches can cover any number of words in either sentence. Once matches are identified, the final alignment is resolved as the largest subset of all matches meeting the following criteria in order of importance:

1. Require each word in each sentence to be covered by zero or one matches.
2. Maximize the number of covered words across both sentences.
3. Minimize the number of chunks (`Ch`), where a *chunk* is defined as a series of matches that is contiguous and identically ordered in both sentences.

4. Minimize the sum of absolute distances between match start positions i and j over all matches. (Break ties by preferring to align words and phrases that occur at similar positions in both sentences.)

While the Meteor aligner is most often used as part of scoring translation quality, it can also be used in other tasks that require rich monolingual phrase alignments. Notably, Meteor is used to create alignments between different systems' translation hypotheses of each source sentence as part of the system combination approach described by Heafield and Lavie (2011).

Meteor Scoring: Given an alignment between hypothesis E' and reference E, the Meteor metric score is calculated as follows. First calculate initial statistics:

- $\langle \mathcal{C}_{\mathtt{f}}(E'), \mathcal{C}_{\mathtt{f}}(E) \rangle$: function words in E' and E. Count any word that appears in the Meteor function word lists estimated from large monolingual data (Denkowski and Lavie, 2011).
- $\langle \mathcal{C}_{\mathtt{c}}(E'), \mathcal{C}_{\mathtt{c}}(E) \rangle$: content words in E' and E. Count any word that does not appear in the function word lists.
- $\langle h_i(\mathcal{C}_{\mathtt{c}}(E')), h_i(\mathcal{C}_{\mathtt{f}}(E')), h_i(\mathcal{C}_{\mathtt{c}}(E)), h_i(\mathcal{C}_{\mathtt{f}}(E)) \rangle$: the number of content and function words in E' and E covered by each type of match h_i. (For example, counts of content and function words covered by exact matches in the hypothesis and reference.)
- $\mathtt{Ch}$: the minimum number of *chunks* (series of matches that are contiguous and identically ordered in both sentences) that the alignment can be divided into.

Calculate weighted precision and recall using match type weights $w_i \in W$ and content-vs-function word weight (δ):

$$\mathcal{P} = \frac{\sum_i \left(w_i \times \left(\delta \times h_i(\mathcal{C}_{\mathtt{c}}(E')) + (1-\delta) \times h_i(\mathcal{C}_{\mathtt{f}}(E')) \right) \right)}{\delta \times \mathcal{C}_{\mathtt{c}}(E') + (1-\delta) \times \mathcal{C}_{\mathtt{f}}(E')} \tag{4.1}$$

$$\mathcal{R} = \frac{\sum_i \left(w_i \times \left(\delta \times h_i(\mathcal{C}_{\mathtt{c}}(E)) + (1-\delta) \times h_i(\mathcal{C}_{\mathtt{f}}(E)) \right) \right)}{\delta \times \mathcal{C}_{\mathtt{c}}(E) + (1-\delta) \times \mathcal{C}_{\mathtt{f}}(E)} \tag{4.2}$$

The harmonic mean of $\mathcal{P}$ and $\mathcal{R}$ parameterized by α (van Rijsbergen, 1979) is then calculated:

$$\mathcal{F}_\alpha = \frac{\mathcal{P} \times \mathcal{R}}{\alpha \times \mathcal{P} + (1-\alpha) \times \mathcal{R}} \tag{4.3}$$

A fragmentation score is calculated using the total number of matched words $\mathtt{M}$ (average over hypothesis and reference) and number of chunks ($\mathtt{Ch}$):

$$\mathtt{M} = \frac{\sum_i \left(h_i(\mathcal{C}_{\mathtt{c}}(E')) + h_i(\mathcal{C}_{\mathtt{f}}(E')) + h_i(\mathcal{C}_{\mathtt{c}}(E)) \right) + h_i(\mathcal{C}_{\mathtt{f}}(E))}{2} \qquad \mathtt{Frag} = \frac{\mathtt{Ch}}{\mathtt{M}} \tag{4.4}$$

The final Meteor score is calculated with fragmentation parameters β and γ:

$$\mathtt{Meteor}(E', E) = \left(1 - \gamma \times \mathtt{Frag}^{\beta} \right) \times \mathcal{F}_\alpha \tag{4.5}$$

Each of the Meteor scoring statistics can be interpreted as a key predictor of post-editing effort. Precision ($\mathcal{P}$) is an inverse measure of the amount of content in the hypothesis that must be *deleted* to match the reference. Recall ($\mathcal{R}$) inversely measures the amount of content that must be *inserted*. Fragmentation ($\mathtt{Ch}$) is a measure of how much *reordering* is required to match the reference. Compared to edit distance-based metrics, Meteor makes a greater distinction between word *choice* and word *order*.

The following metric parameters can be tuned to maximize agreement between Meteor scores and human assessments of translation quality:

- $W = \langle w_i, ..., w_n \rangle$: an individual weight for each type of match, allowing distinctions such as penalizing a stem match, which likely requires editing for grammaticality, more harshly than a synonym match, which may not require editing. There are currently 4 weights: exact, stem, synonym, and paraphrase. The weight for exact matches is fixed at 1.
- α: the balance between precision and recall, allowing greater penalties for either insertion or deletion requirements.
- β, γ: the weight and severity of fragmentation, allowing fine-tuning the cost for reordering text.
- δ: the relative contribution of content versus function words, allowing greater penalties for important words that tend to be more difficult to translate.

Parameter Optimization: Tuning a version of Meteor to approximate a given evaluation task requires a set of n MT outputs with reference translations plus a set of human-annotated scores $Y = \langle y_1, ..., y_n \rangle$ (quality scale assessments, HTER scores, or other numerical scores). Meteor scores the MT outputs, producing metric scores $X = \langle x_1, ..., x_n \rangle$. Ideally, X should be strongly correlated with Y, meaning that a high metric score should correspond to a high human score and vice versa. We measure correlation with Pearson's correlation coefficient r:

$$r(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{X})(y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{Y})^2}} \tag{4.6}$$

where $\bar{X}$ and $\bar{Y}$ are means. During tuning, sufficient statistics are calculated for each MT output, allowing it to be rapidly re-scored with various parameter settings. We then conduct an exhaustive parametric sweep over feasible parameter values to maximize r between X and Y over all MT outputs. This guarantees globally optimal metric parameters for the data set. To account for length variation in training data, we weight each score x_i or y_i by the length of the *reference* translation when calculating r.

4.2.2 Evaluation Experiments

Meteor has been successfully tuned to replicate several types of human quality judgments. The most widely used "ranking" version of Meteor (Denkowski and Lavie, 2011) is shown to reliably assign higher scores to the types of translations preferred by human annotators in WMT evaluations in a variety of languages (Callison-Burch et al., 2012). A version tuned to numerical adequacy scale scores (Denkowski and Lavie, 2010b) shows good linear correlation with this type of human judgment for English (Callison-Burch et al., 2010)[2]. Current work focuses on using Meteor to predict human post-editing effort.

Initial work investigating Meteor's ability to predict post-editing effort uses human-targeted translation edit rate (HTER) (Snover et al., 2006) judgments from the DARPA Global Autonomous Language Exploitation (GALE) project (Olive et al., 2011). We use two sets of HTER scores calculated from post-editing translations into English in two phases of the project: P2 and P3. For each data set, we tune a version of Meteor to maximize the sentence-level length-weighted Pearson's correlation coefficient r. We evaluate Meteor's ability to *fit* the data by measuring correlation on the same data set and ability to *generalize* to other HTER data by measuring correlation on the alternate data set. We evaluate the current version of Meteor (1.3) and a version without content-vs-function word distinction (1.2), comparing them against baseline metrics BLEU, TER, and Meteor version 1.0 (defined in §1.3.3) that are not tuned on HTER data.

[2] http://www.itl.nist.gov/iad/mig/tests/metricsmatr/2010/results/correlation_English_Adequacy7Average_segment.html

Metric	Tuning Data	P2 r	P3 r
BLEU	N/A	-0.545	-0.489
TER	N/A	0.592	0.515
Meteor (1.0)	N/A	-0.625	-0.568
Meteor (1.2)	P2	-0.640	-0.596
	P3	-0.638	-0.600
Meteor (1.3)	P2	-0.642	-0.594
	P3	-0.625	-0.612

Table 4.1: Correlation of metric scores with HTER.

Task	α	β	γ	δ	w_{exact}	w_{stem}	w_{syn}	w_{par}
WMT Ranking	0.85	0.20	0.60	0.75	1.00	0.60	0.80	0.60
GALE HTER	0.40	1.50	0.35	0.55	1.00	0.20	0.60	0.80
System Tuning	0.50	1.00	0.50	0.50	1.00	0.50	0.50	0.50

Table 4.2: Comparison of Meteor 1.3 parameters for different tasks.

The correlation results of these experiments are shown in Table 4.1 while the optimal metric parameters for Meteor 1.3 are shown in Table 4.2. Although the current version of Meteor outperforms all baseline metrics, the version without word type distinction actually performs best on alternate data sets. This, along with the contrast between optimal parameters for HTER and WMT ranking reveal shortcomings in the formulation of HTER. As HTER makes no distinction between content and function words, δ is near 0.5. As identical base words with different inflections are treated as non-matches by TER, the weight for stem matches is near 0. Whereas these parameters allow greater distinctions to be made for adequacy and ranking data, they only provide additional possibilities of overfitting for HTER data. These results highlight the need for more accurate types of editing measures to train metrics to better predict post-editing as discussed in §4.3 and Chapter 5.

4.2.3 MT System Optimization Experiments

Engineering automatic metrics for optimization versus evaluation presents several new challenges. The types of sentences encountered in k-best lists, especially during early optimizer iterations, are vastly different from the top-best output of systems tuned to BLEU. Translation hypotheses will often receive very low scores and many candidates may have similar or identical scores despite differences in word choice and order. A metric must navigate this space of translations, correctly selecting better translations to guide the optimization algorithm. Algorithms such as MERT that optimize courpus-level metric scores present a particular risk of overfitting. For example, the ranking version of Meteor places several times more weight on recall than precision, fitting the human preference displayed in WMT evaluations. However, tuning a system to this version can produce artificially long sentences with high recall but low precision and high fragmentation. As such translations are generally not present in the output of BLEU-tuned systems in WMT, tuned versions of Meteor have difficulty learning this distinction.

To address these difficulties, we have designed a "tuning" version of Meteor, also listed in Table 4.2. This version is more balanced between precision and recall, content and function words, and word choice and order. We evaluate this version of Meteor against the ranking version and standard BLEU in two translation scenarios. The first high-resource scenario uses the WMT11 (Callison-Burch et al., 2011) French–English data set (approximately 14 million bilingual sentences and 1.2 billion monolingual words) while

French–English			
Tuning Metric	BLEU	TER	Meteor-rank
BLEU	28.27	53.94	54.07
Meteor-rank	27.05	56.30	54.44
Meteor-tune	28.14	54.14	54.11

Urdu–English			
Tuning Metric	BLEU	TER	Meteor-rank
BLEU	23.67	72.48	50.45
Meteor-rank	19.28	94.64	49.78
Meteor-tune	24.89	69.54	51.29

Table 4.3: Metric scores for phrase-based translation systems tuned to various objective functions.

the second low-resource scenario uses the NIST2009 OpenMT (Przybocki, 2009) Urdu–English data set (approximately 87 thousand bilingual sentences and 900 million monolingual words). For each scenario, we build a standard phrase-based statistical translation system using the Moses toolkit (Koehn et al., 2007) and conduct 3 independent optimization runs for each tuning metric using the Z-MERT implementation (Zaidan, 2009) of minimum error rate training (Och, 2003). The average metric scores over optimizer runs are reported in Table 4.3. In the high-resource scenario where BLEU is generally stable, we see comparable performance from Meteor-tune. However, in the low-resource scenario where BLEU has more difficulty discriminating between low-scoring translations, Meteor-tune shows significant improvement across metrics. In both scenarios, the tuning version of Meteor significantly outperforms the ranking version.

4.3 Proposed Work: Meteor for Optimizing Online Post-Editing Systems

Our experiments with Meteor focus on both evaluation and optimization. Given the mismatch between HTER's coarse approximation and Meteor's descriptive features, we will use several types of data collected from human translators (discussed in §5.4) to learn metric metric parameters for more accurate measures. We will fit Meteor scores to the following types of human data:

- Human-targeted translation edit rate (HTER): the most widely used measure of post-editing for comparison to existing data sets.
- Human post-assessments: translator assessment of how much work was just required.
- Keystrokes: a measure of the amount of mechanical effort required to edit translations.
- Editing time: a measure of the "bottom line" time cost of a translation project that uses post-editing.
- Average pause ratio (Lacruz et al., 2012): an approximation of cognitive effort required to edit translations.

Following Snover et al. (2009) and Denkowski and Lavie (2010a), we will examine the optimal metric parameters and correlation for each types of measure. Metric parameters provide insight into what aspects of translation quality are most important for reducing demand on human editors. Correlation across data sets of the same type (cross-validation) show the *stability* of a measure. An ideal measure should prefer the same types of translations across data sets, providing a stable target for translation quality. Evaluation experiments will be carried out in full for at least one language scenario. Other scenarios will be added as resources allow.

The move from MERT to PRO or large margin learning also moves from corpus-level optimization to sentence-level optimization, providing an opportunity for sentence-level metrics such as Meteor to yield better results than approximations of sentence-level BLEU. In addition to tuning to BLEU, we will tune an extended feature set system to the tuning version of Meteor plus versions tuned to post-editing effort. If resources allow, we will conduct a comparison of BLEU and Meteor-tuned systems with human translators.

This will evaluate the degree to which Meteor can serve as a sufficient proxy for editing effort assessment during optimization. Ideally, a metric should be able to consistently predict post-editing effort during both optimization and evaluation, ultimately leading to a system configuration that is best for human translators.

Chapter 5

Post-Editing Data: Feedback and Analysis

While translation model adaptation, optimization, and evaluation experiments can all be carried out with simulated data, the final systems produced are intended for actual human post-editing scenarios. Conducting real-time translation experiments that evaluate the effectiveness of our work and yield valuable post-editing data requires an interactive translation environment that allows translators to work directly with our translation systems. We have developed a lightweight web-based translation editing environment termed *TransCenter* (a shortened form of "Translation Center") that allows us to present translations from our system to human translators and collect detailed post-editing statistics. In addition to evaluating the effectiveness of our systems for post-editing, collecting this data will facilitate a deeper statistical analysis of translation post-editing. In §5.3, we describe an initial set of experiments with human translators that examines the ability of existing measures to predict editing effort. The results of this initial work highlights the need for more accurate data collection and descriptive measures of editing.

5.1 Related Work

Software frameworks that bring machine translation into human translation workflows are generally referred to as computer-aided translation (CAT) tools. In general, CAT tools aim to pair support for technologies such as translation memories with support for familiar file types such as Microsoft Word Documents and HTML web pages. The most widely used CAT tool is the commercially developed SDL Trados software suite[1] that provides support for translation memories, terminology dictionaries, and plug-ins for machine translation engines. This software runs on desktop computers and is limited to Microsoft Windows. Recently, the natural language processing research community has developed several open source CAT tools. Penkale and Way (Penkale and Way, 2012) introduce the SmartMATE online translation editing environment that supports translation memories, glossaries, and machine translation and accepts a variety of input file formats. SmartMATE also features the ability to train new MT systems from existing translation memories. Aziz et al. (2012) introduce PET (Post-Editing Tool), a standalone desktop application that provides a simple interface for editing MT output and records data such as editing time, keystrokes, and translator assessments. Ortiz-Martínez et al. (2012) describe the translation workbench software developed as part of the CASMACAT[2] (Cognitive Analysis and Statistical Methods for Advanced Computer Aided Translation) project. The project aims to develop a full-featured open source translator's workbench environment that can be accessed via the web or installed locally. This project also focuses on deeper cognitive analysis of translation editing. Completed work includes experiments showing that a simple, streamlines user interface facilitates faster editing than a complex interface that offers more options (Alabau et al., 2012).

[1] http://www.trados.com/en/
[2] http://casmacat.eu/

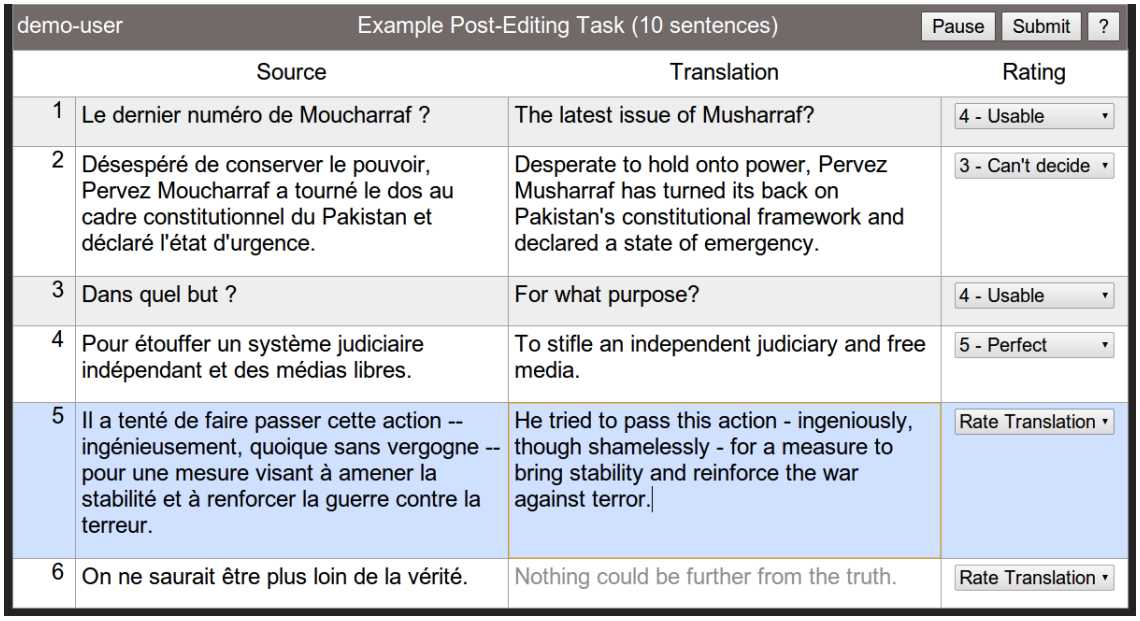


Figure 5.1: TransCenter web-based translation editor

5.2 Completed Work: TransCenter Software

In contrast to CAT tools that aim to provide complete translation editing solutions, our TransCenter software (Denkowski and Lavie, 2012b) focuses on highly accurate translation editing *data collection*. Running the platform-independent TransCenter server provides a simple web-based translation editing environment that translators can access from any computer with a web browser. The browser-based editor interface displays source sentences and translations in a two-column format that emphasizes clarity and simplicity. Fast keyboard navigation and automatic highlighting provide a smooth user experience. During editing, all user actions (key presses and mouse clicks) are logged so that the full translation editing process can be replayed and analyzed. After each sentence is edited, translators are immediately asked to rate the amount of work they feel was required, yielding maximally accurate feedback. TransCenter also records the number of seconds each sentence is focused, allowing for exact timing measurements. A pause button is available if translators need to take breaks. TransCenter can generate reports of translation and post-editing effort as measured by (1) keystroke, (2) exact timing, (3) actual translator post-assessment. Final translations are also available for calculating edit distance. TransCenter server is written in Python and can be run on Linux, Microsoft Windows, and Mac OS. The web-based interface supports major browsers including Google Chrome, Mozilla Firefox, and Microsoft Internet Explorer. All data is stored in open formats for maximum interoperability. A screenshot of the web-based translation editor is shown in Figure 5.1 and editing reports are shown in Figures 5.2 and 5.3.

Where CAT tools aim for feature richness, we aim for simplicity and accuracy. In designing TransCenter, we intentionally avoid dealing with resources such as translation memories and dictionaries and file formats that contain complex document formatting. While helpful for translators, suggestions from memories and dictionaries complicate analysis. The availability of fast matches from existing translations to correct translation errors can mask deficiencies in MT output. Additionally, support for document formatting complicates cognitive effort analysis. If a translator spends 5 minutes editing a sentence, it is unclear what percentage of that time went toward fixing translation errors versus formatting the target document to match the source style. To maximize the accuracy of collected data, TransCenter operates only on plain text

ID	Source	Translation	Edited	Rating	Keypress	Mouseclick	Edits	Time
1	Le dernier numéro de Moucharraf ?	The latest issue of Musharraf?	Musharraf's last act?	2	58	1	17	37854
2	Désespéré de conserver le pouvoir, Pervez Moucharraf a tourné le dos au cadre constitutionnel du Pakistan et déclaré l'état d'urgence.	Desperate to hold onto power, Pervez Musharraf has turned its back on Pakistan's constitutional framework and declared a state of emergency.	Desperate to hold onto power, Pervez Musharraf has discarded Pakistan's constitutional framework and declared a state of emergency.	4	63	2	11	25482
3	Dans quel but ?	For what purpose?	His goal?	4	16	3	13	1345493524384
4	Pour étouffer un système judiciaire indépendant et des médias libres.	To stifle an independent judiciary and free media.	To stifle the independent judiciary and free media.	2	30	2	2	1345493531670
5	Il a tenté de faire passer cette action -- ingénieusement, quoique sans vergogne -- pour une mesure visant à amener la stabilité et à renforcer la guerre contre la terreur.	He tried to pass this action - ingeniously, though shamelessly - for a measure to bring stability and reinforce the war against terror.	He tried to pass this action - ingeniously, though shamelessly - for a measure to bring stability and reinforce the war against terror.	3	46	5	49	2690987083857
6	On ne saurait être plus loin de la vérité.	Nothing could be further from the truth.	Nothing could be further from the truth.	4	13	2	0	1345493555451

Figure 5.2: TransCenter document editing summary report

Time	Sentence 1 Edits
Initial	The latest issue of Musharraf?
1345076800710	The latest issue of Musharraf?
1345076800861	The la of Musharraf?
1345076801036	The las of Musharraf?
1345076801212	The last of Musharraf?
1345076801341	The last of Musharraf?
1345076801508	The last a of Musharraf?
1345076801644	The last ac of Musharraf?
1345076801852	The last act of Musharraf?
1345076810117	The last act of Musharraf?
1345076811637	The of Musharraf?
1345076811796	The ofMusharraf?
1345076811973	The oMusharraf?
1345076814613	The Musharraf?
1345076815893	Musharraf'?
1345076816100	Musharraf's?
1345076816269	Musharraf's ?
1345076816535	Musharraf's last act?
Final	Musharraf's last act?

Figure 5.3: TransCenter sentence edit history

with a single machine translation hypothesis per source sentence. While this may lead to additional work for translators working with TransCenter, the highly accurate data resulting from translation editing can be used to improve the quality of underlying translation systems. The improved systems can then be plugged in as resources for feature-rich CAT tools to ultimately reduce the load placed on human translators. In this sense, our work on TransCenter is largely complementary to work on CAT tools.

5.3 Completed Work: Examination of Machine Translation for Post-Editing as a Task

Large machine translation evaluation campaigns such as the ACL Workshops on Statistical Machine Translation (Callison-Burch et al., 2011) and NIST Open Machine Translation Evaluations (Przybocki, 2009) focus on improving translation adequacy, the perceived quality of fully automatic translations compared to

	Rank	HTER	Translation
Reference	–	–	Only the crème de la crème of the many applicants will fly to the USA.
System 1	$1st$	0.40	Only the crème de la crème from many candidates, it's going to go to the US.
Post-edit 1	–	–	Only the crème de la crème from many candidates will fly to the US.
System 2	$2nd$	0.20	Only crème de la crème of many customers will travel to the US.
Post-edit 2	–	–	Only the crème de la crème of many applicants will fly to the US.

	BLEU	HTER	Translation
Reference	–	–	The problem is that life of the lines is two to four years.
System 1	0.49	0.29	The problem is that life is two lines, up to four years.
Post-edit 1	–	–	The problem is that life of the lines is two to four years.
System 2	0.34	0.14	The problem is that the durability of lines is two or four years.
Post-edit 2	–	–	The problem is that the life of lines is two to four years.

Table 5.1: Cases where lower-ranked (top) or lower BLEU-scoring (bottom) MT outputs require less work to post-edit. Lower HTER indicates fewer edit operations required.

reference translations. As such, current techniques for MT system building, optimization, and evaluation are largely geared toward improving performance on this task. Originally introduced by the Linguistics Data Consortium, adequacy ratings elicit straightforward quality judgments of machine translation output according to numeric scales (LDC, 2005). Recent WMT evaluations (Callison-Burch et al., 2007; Callison-Burch et al., 2011) use *ranking*-based evaluation to abstract away from concepts such as adequacy and grammaticality as well as difficult-to-decide numeric ratings. Human judges are simply asked to rank several MT outputs for the same sentence from best to worst according to a reference translation. It is left up to judges to determine the relative severity of different types of translation errors when comparing translations. Whereas MT systems targeting adequacy should maximize the semantic similarity of automatic translations with reference translations, systems targeting post-editing utility should minimize the effort required by human translators to correct automatic translations. This is most often measured by cased human-targeted translation edit rate (HTER) (Snover et al., 2006) that depends on alignments from the TER metric.

5.3.1 Translation Evaluation Examples

The adequacy and post-editing tasks bear some similarities, as automatic translations that have high similarity to reference translations often require minimal post-editing. However, when MT outputs contain errors, the most adequate translations are often not the easiest to post-edit. Table 5.1 shows two examples from the difficult Czech-to-English translation track of the 2011 EMNLP Workshop on Statistical Machine Translation (Callison-Burch et al., 2011) with minimally post-edited translations and HTER scores. In the first case, the translation deemed more adequate by expert judges actually requires more effort to post-edit. In the second example, sentence 2 is penalized by BLEU for using a different word order from the reference even though it is both more adequate and less work to correct. These examples illustrate types of errors that have a large impact on sentence meaning but require relatively little work to correct, as well as accumulated minor errors that do not impact meaning, but are cumbersome to correct.

5.3.2 Initial Post-Editing Experiments

To empirically evaluate the behavior of human post-editors and the effectiveness of current MT techniques for predicting translation utility, we conduct a series of experiments to simulate a real-world localization

Reference	BLEU	TER	Meteor
Gold	0.32	0.49	0.58
Post-edit	0.79	0.12	0.90
Post-edit vs Gold	0.34	0.48	0.59

Table 5.2: Corpus-level BLEU, TER, and Meteor scores of MT output against gold standard and post-edited references.

scenario (Denkowski and Lavie, 2012a).

Data Collection: We select 5 English–Spanish bilingual documents in the software documentation domain, totaling 90 sentences. Each English sentence is translated into Spanish using two translation engines: Microsoft Translator's online service[3] and a phrase-based Moses system representative of the 2011 WMT baseline (Koehn et al., 2007; Callison-Burch et al., 2011). The outputs of the two systems, which have significant lexical differences but are statistically indistinguishable by automatic metrics, are combined into a single data set of 180 translations. The Spanish side of the bilingual data provides reference translations. We employ the assistance of an expert translator and several students of translation studies from Kent State University's Institute for Applied Linguistics[4] to obtain two types of annotation for each automatic translation. First, each translation is post-edited by 2 bilingual student translators from a pool of 7 total. Second, the expert translator assigns a rating from 1 to 4 reflecting the degree of post editing that appears to have been carried out:

1. No editing required
2. Minor editing, meaning preserved
3. Major editing, meaning lost
4. Re-translation

Metric Experiments: We use the BLEU (Papineni et al., 2002), TER (Snover et al., 2006), and Meteor (Denkowski and Lavie, 2011) metrics to evaluate MT outputs against two types of reference translations. In Table 5.2, "gold" refers to the gold standard references from the bilingual data (unseen by human editors). "Post-edit" refers to using the post-edited reference with the minimum edit distance to each MT output, as in HTER scoring. Post-edited lines correspond to HTER, "H-BLEU", and "H-Meteor". Finally, the last line scores the post-edited reference set against the gold standard. The significantly lower metric scores against gold standard references compared to the post-edited references illustrate the known problem that metrics are good at detecting similar translations, but poor at evaluating sentences with different structure and lexical choice. Additionally, metrics assign nearly the same scores to the closest post-edited references as they do to the raw MT outputs, though in the case of Meteor, both scores are relatively high, reflecting more accurate evaluation.

In a second experiment, we simplify the 4-point predictions into two groups: usable (1 and 2) and non-usable (3 and 4), corresponding to whether the expert translator would personally post-edit or re-translate each sentence. MT outputs were largely deemed to be usable (90.6%) with a minority classified as non-usable (9.4%). We examine the distributions of sentence-level BLEU and HTER scores for each group. As shown in Figure 5.4, the BLEU score distributions overlap completely and translations are clustered

[3] http://www.microsofttranslator.com/
[4] http://appling.kent.edu/

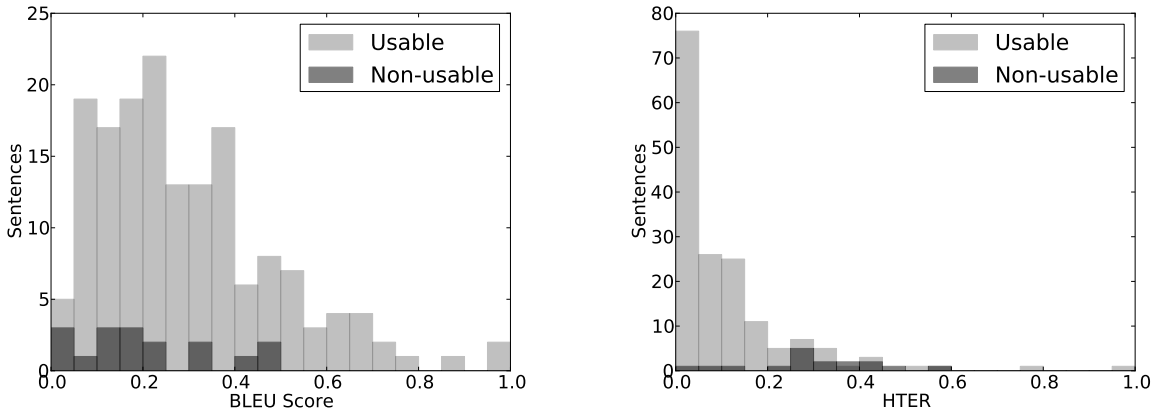


Figure 5.4: Distributions of BLEU scores and HTER for usable and non-usable translations.

in the same region. No quality threshold (visualized as a vertical line on the graph) can reliably separate usable from non-usable translations. The HTER scores show that an expert is largely able to detect easily correctable translations, judging nearly all translations with HTER under 0.2 to be usable. Above 0.2, translations requiring comparable numbers of edits are judged to be both usable and non-usable. These results illustrate that not only are standard metrics such as BLEU poor at discriminating between easy and difficult to post-edit translations, but measures of post-editing themselves do not always agree. As HTER and human post-assessments are both removed from the intermediate editing process, the resulting scores are not always accurate.

5.4 Proposed Work: Real-Time Translation Analysis

Extensions to TransCenter: Currently, TransCenter only supports pre-generated translations for all input sentences. Accommodating real-time translation experiments requires tighter integration with the underlying MT system. Using the `pycdec` (Chahuneau et al., 2012) interface, we will fully integrate the `cdec` (Dyer et al., 2010) aligner, grammar extractor, and decoder with TransCenter, allowing new bilingual sentence pairs created by human translators to be added to the model as new data is translated. This works with the support for multiple translation contexts that we will add to the grammar extractor, enabling multiple translators to work at the same time without interference. These extensions require significant additions to the client and server code to allow asynchronous communication between the user interface and underlying MT engine.

We will use the updated version of TransCenter to conduct a series of real-time translation editing experiments. In all experiments, TransCenter will collect the number of seconds each translator maintains focus of each sentence and what keystrokes occur during that span (with timestamps). TransCenter will also elicit a 5-point judgment of translation difficulty immediately after each sentence is translated and calculate edit distance between the MT output and final post-edited translation. Additionally, we will extend TransCenter to automatically generate the following *normalized* statistics using the base data collected:

- Translation time: number of seconds a translation is focused divided by either the number of words or number of characters in the source sentence (both versions tested)
- Mechanical effort: number of keystrokes while a translation is focused divided by either the number

of words or number of characters in the source sentence (both versions tested)

- Average pause ratio (APR) (Lacruz et al., 2012): a measure based on the idea that the number and length of pauses made during editing reflect the cognitive effort put forth by a translator prior to certain editing actions. More difficult edits tend to require longer pauses. Given the total time editing a sentence $\mathcal{T}(E')$, number of words in the sentence $|E'|$, total time in pause $\mathcal{T}(\texttt{Pause}(E'))$, and number of pauses in a sentence $|\texttt{Pause}(E')|$, APR is calculated as follows. First calculate the components:

$$\texttt{TimePerWord}(E') = \frac{\mathcal{T}(E')}{|E'|} \qquad \texttt{TimePerPause}(E') = \frac{\mathcal{T}(\texttt{Pause}(E'))}{|\texttt{Pause}(E')|} \tag{5.1}$$

  Average pause ratio is then defined:

$$\texttt{APR}(E') = \frac{\texttt{TimePerPause}(E')}{\texttt{TimePerWord}(E')} \tag{5.2}$$

Data Collection: We will use the extended version of TransCenter to present translations from our systems to human editors. Editors will be a combination of student translators from the Kent State University Institute for Applied Linguistics and professional translators as resources allow. Translators will be given directions to edit normally to produce "good enough" translations and not scrutinize over minor details so long as the translation is grammatical and clearly conveys the meaning of the source sentence. Translators will know that sentences are machine translated, but will not be told specifics about systems (on-demand versus online translation models or basic versus extended feature sets).

Data Analysis: Experiments will focus on collecting sufficient data to determine whether our online, extended feature set translation system yields significant productivity gains over a baseline. We will at minimum collect sufficient data to compare the on-demand baseline system to our best performing online system in both news and TED talk domains for one language direction. As resources allow, we will also include an intermediate baseline that incorporates post-edit feedback and uses the online versions of the basic feature set plus the post-edit support indicator feature. This baseline will allow us to measure the impact specifically of our extended feature set. We will also expand to other language scenarios as human translators are available. To account for variation in translator speed and tendency to over or under-edit, we will have a calibration or "burn-in" stage where each participant first edits translations from the baseline system for a fixed set of sentences before moving to translations of new documents from various systems. The common burn-in data will allow us to measure *relative* changes in the various statistics gathered by TransCenter when editing output from our different translation models. Once this data is gathered, we will conduct a comprehensive analysis to measure the impact of our extended translation systems on all recorded measures of post-editing effort.

We will also conduct a significant analysis of different post-edit measures. As discussed in §4.3, we will tune versions of Meteor for each type of assessment collected. Examining metric parameters will allow us to see what attributes of translation quality are favored by a measure. Comparing correlation values in cross-validation and between different measures will provide insight into the stability of the measures. As in §5.3.2, we will examine the distributions of scores to see where measures agree and disagree. Finally, we will experiment with *combining* different measures to see if certain combinations lead to improved stability. As all statistics are calculated automatically by TransCenter, sophisticated measure combinations can be calculated for system evaluation and metric tuning.

(Possible Experiment) Commercial Data: It is likely that we will have access to post-edited data in a large number of target languages from a commercial provider. If this data becomes available, we will be able

to tune post-editing versions of Meteor for a wide range of languages. Examining metric parameters and generalizability will yield valuable information as to how post-editing tasks are related across languages.

Chapter 6

Summary and Timeline

6.1 Summary

Our work focuses on improving the usability of machine translation for human post-editing by addressing three key areas: translation modeling, automatic metrics, and data collection. The key points from each section are described below. Table 6.1 lists the status of each major task within these three broader areas.

Online Learning for Machine Translation: We cast MT for post-editing as an online learning task where new training instances are created as humans edit system output (§3). We present an online extension of the powerful on-demand hierarchical phrase-based translation model with suffix array-backed features (§3.2). We discuss an extended feature set that allows this model to learn from multiple translation contexts over time and encodes all model adaptation in sentence-level grammars (§3.3). This allows our model to be optimized using standard methods such as MERT and PRO.

Translation System Optimization and Evaluation for Post-Editing Tasks: We describe the Meteor metric for automatic evaluation and optimization of MT systems (§4). Meteor creates a flexible alignment between hypothesis-reference pairs and calculates several scoring statistics that are interpretable as measuring of post-editing effort (§4.2.1). Scoring parameters can be optimized to fit various types of human judgments. Meteor can be used to optimize translation systems in scenarios where BLEU breaks down, select optimal system configurations for post-editing, and provide insight into the properties of translation quality that are most important for minimizing editing effort (§4.3).

Real-Time Post-Editing Data Collection and Analysis: We describe a series of experiments for collecting detailed post-editing data (§5). We present the TransCenter web-based framework for collecting several types of highly accurate data from human translators (§5.2). We discuss MT for post-editing as a distinct task and present the results of initial post-editing experiments. The results show that (1) Meteor is significantly better than standard metrics at predicting post-editing effort and (2) even "gold standard" measures of post-editing can be inaccurate (§5.3). We finally outline a set of experiments for collecting valuable data that will be used to evaluate the impact of our online translation models and optimization metrics on human editing requirements.

6.2 Contributions

Our work will contribute the following to the fields of human and machine translation:

Translation Modeling	
Online grammar extraction	Completed
Multiple context feature sets	Proposed
Data size feature sets	Proposed
Automatic Metrics	
Extended version of Meteor metric	Completed
Initial Meteor optimization and evaluation for adequacy	Completed
Meteor evaluation for post-editing	Proposed
Meteor optimization for post-editing	Proposed
Data Collection	
Basic version of TransCenter	Completed
Initial analysis of adequacy versus post-editing tasks	Completed
Initial post-editing data analysis	Completed
Extended version of TransCenter with live MT and feedback	Proposed
Real-time post-editing experiments with TransCenter	Proposed
Full post-editing data evaluation and analysis	Proposed

Table 6.1: Progress report of completed and proposed work.

- An online translation model that learns non-local reordering as well as new translations, uses a powerful suffix array-backed feature set, and can be optimized using standard MERT.
- An extended translation feature set that allows standard PRO to learn to trust different sources of feedback over time.
- Automatic metrics specifically tailored to optimize and evaluate translation models targeting post-editing usability.
- An in-depth analysis of various types of post-editing measures and measure combinations to determine the most reliable method for evaluating human editing effort.
- A focused study on the impact of online learning on human post-editing requirements.

As a result of our work, the following software tools will be made freely available under open source licenses:

- An implementation of online grammar extraction with extended feature sets, integrated with the `cdec` toolkit through `pycdec`.
- An extended version of the Meteor metric for optimization and evaluation of MT systems targeting post-editing tasks.
- A fully interactive version of the TransCenter web-based translation environment that supports live translation and feedback.

As a result of our experiments, the following data will be made freely available:

- All bilingual text and post-edited translations generated during data collection.
- Data files for the various editing statistics collected during data collection.

6.3 Timeline

The proposed work is scheduled on a 12 month timeline ending May 2014. An intermediate checkpoint is included for November 2013 when the first end-to-end post-editing experiments with the extended version of the online translation system are scheduled to be finished. Tasks are organized into blocks of two months each as follows.

June–July 2013

- Multiple context feature sets for online grammar learning (inc. simulated experiments)
- Data size feature sets for online grammar learning (inc. simulated experiments)

August–September 2013

- Extended version of TransCenter with live MT and feedback
- Start real-time post-editing experiments

October–November 2013

- Continue real-time post-editing experiments
- Meteor evaluation for post-editing as data arrives (inc. experiments)
- **Checkpoint: report first end-to-end human post-editing experiments**

December 2013–January 2014

- Meteor optimization for post-editing (inc. experiments)
- Post-editing data evaluation and analysis as data is available

February–March 2014

- Additional translation experiments and analysis for as many language scenarios as resources allow

April–May 2014

- Writing thesis document
- **Checkpoint: thesis defense**

References

[Agarwal and Lavie2008] Abhaya Agarwal and Alon Lavie. 2008. Meteor, M-BLEU and M-TER: Evaluation metrics for high-correlation with human rankings of machine translation output. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 115–118, Columbus, Ohio, June. Association for Computational Linguistics.

[Alabau et al.2012] Vicent Alabau, Luis A. Leiva, Daniel Ortiz-Martínez, and Francisco Casacuberta. 2012. User evaluation of interactive machine translation systems. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 20–23.

[Axelrod et al.2005] Amittai Axelrod, Ra Birch Mayne, Chris Callison-burch, Miles Osborne, and David Talbot. 2005. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In *Proceedings of the 2005 International Workshop on Spoken Language Translation*.

[Aziz et al.2012] Wilker Aziz, Sheila Castilho Monteiro de Sousa, and Lucia Specia. 2012. PET: a tool for post-editing and assessing machine translation. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*.

[Banerjee and Lavie2005] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June. Association for Computational Linguistics.

[Bannard and Callison-Burch2005] Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 597–604, Ann Arbor, Michigan, June. Association for Computational Linguistics.

[Blain et al.2011] Frédéric Blain, Jean Senellart, Holger Schwenk, Mirko Plitt, and Johann Roturier. 2011. Qualitative analysis of post-editing for high quality machine translation. In *Proceedings of the twelfth Machine Translation Summit International Association for Machine Translation*.

[Brown et al.1993] Peter Brown, Vincent Della Pietra, Stephen Della Pietra, and Robert Mercer. 1993. The mathematics of statistical Machine Translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

[Callison-Burch et al.2005] Chris Callison-Burch, Colin Bannard, and Josh Schroeder. 2005. Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 255–262, Ann Arbor, Michigan, June. Association for Computational Linguistics.

[Callison-Burch et al.2006] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of bleu in machine translation research. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*.

[Callison-Burch et al.2007] Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2007. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June. Association for Computational Linguistics.

[Callison-Burch et al.2010] Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, Uppsala, Sweden, July. Association for Computational Linguistics. Revised August 2010.

[Callison-Burch et al.2011] Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland, July. Association for Computational Linguistics.

[Callison-Burch et al.2012] Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, June. Association for Computational Linguistics.

[Cer et al.2010] Daniel Cer, Christopher D. Manning, and Daniel Jurafsky. 2010. The best lexical metric for phrase-based statistical mt system optimization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 555–563, Los Angeles, California, June. Association for Computational Linguistics.

[Cettolo et al.2012] Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit^3: Web inventory of transcribed and translated talks. In *Proceedings of the Sixteenth Annual Conference of the European Association for Machine Translation*.

[Chahuneau et al.2012] Victor Chahuneau, Noah A. Smith, and Chris Dyer. 2012. pycdec: A python interface to cdec. *The Prague Bulletin of Mathematical Linguistics*, 98:51–61.

[Chen and Goodman1996] Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 310–318, Santa Cruz, California, USA, June. Association for Computational Linguistics.

[Chiang2007] David Chiang. 2007. Hierarchical phrase-based translation. 33.

[Clark et al.2011] Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA, June. Association for Computational Linguistics.

[Clark et al.2012] Jonathan Clark, Alon Lavie, and Chris Dyer. 2012. One system, many domains: Open-domain statistical machine translation via feature augmentation. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas*.

[Clark2012] Jonathan Clark. 2012. Locally non-linear learning via feature induction in statistical machine translation. In *Thesis Proposal, Carnegie Mellon University*, April.

[Daumé III2004] Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at http://pub.hal3.name#daume04cg-bfgs, implementation available at http://hal3.name/megam/, August.

[Daume III2007] Hal Daume III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic, June. Association for Computational Linguistics.

[Denkowski and Lavie2010a] Michael Denkowski and Alon Lavie. 2010a. Choosing the right evaluation for machine translation: an examination of annotator and automatic metric performance on human judgment tasks. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*.

[Denkowski and Lavie2010b] Michael Denkowski and Alon Lavie. 2010b. METEOR-NEXT and the METEOR paraphrase tables: Improved evaluation support for five target languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 339–342, Uppsala, Sweden, July. Association for Computational Linguistics.

[Denkowski and Lavie2011] Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 85–91, Edinburgh, Scotland, July. Association for Computational Linguistics.

[Denkowski and Lavie2012a] Michael Denkowski and Alon Lavie. 2012a. Challenges in predicting machine translation utility for human post-editors. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas*.

[Denkowski and Lavie2012b] Michael Denkowski and Alon Lavie. 2012b. TransCenter: Web-based translation research suite. In *AMTA 2012 Workshop on Post-Editing Technology and Practice Demo Session*.

[Dyer et al.2010] Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12, Uppsala, Sweden, July. Association for Computational Linguistics.

[Dyer et al.2013] Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

[Eidelman2012] Vladimir Eidelman. 2012. Optimization strategies for online large-margin learning in machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 480–489, Montréal, Canada, June. Association for Computational Linguistics.

[Francisco-Javier López-Salcedo and Casacuberta2012] Germán Sanchis-Trilles Francisco-Javier López-Salcedo and Francisco Casacuberta. 2012. Online learning of log-linear weights in interactive machine translation. pages 277–286.

[Galley et al.2004] Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

[Habash et al.2009] Nizar Habash, Owen Rambow, and Ryan Roth. 2009. MADA+TOKAN: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, April. The MEDAR Consortium.

[Hardmeier et al.2012] Christian Hardmeier, Joakim Nivre, and Jörg Tiedemann. 2012. Tree kernels for machine translation quality estimation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 109–113, Montréal, Canada, June. Association for Computational Linguistics.

[Hardt and Elming2010] Daniel Hardt and Jakob Elming. 2010. Incremental re-training for post-editing smt. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*.

[He et al.2010] Yifan He, Yanjun Ma, Johann Roturier, Andy Way, and Josef van Genabith. 2010. Improving the post-editing experience using translation recommendation: A user study. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*.

[Heafield and Lavie2011] Kenneth Heafield and Alon Lavie. 2011. CMU system combination in WMT 2011. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 145–151, Edinburgh, Scotland, United Kingdom, 7.

[Heafield2011] Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, July. Association for Computational Linguistics.

[Hopkins and May2011] Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

[Knight1999] Kevin Knight. 1999. A statistical MT tutorial workbook, August. Unpublished.

[Koehn et al.2003] Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of NAACL/HLT 2003*.

[Koehn et al.2007] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

[Koponen2012] Maarit Koponen. 2012. Comparing human perceptions of post-editing effort with post-editing operations. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 181–190, Montréal, Canada, June. Association for Computational Linguistics.

[Lacruz et al.2012] Isabel Lacruz, Gregory M. Shreve, and Erik Angelone. 2012. Average Pause Ratio as an Indicator of Cognitive Effort in Post-Editing: A Case Study. In *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, pages 21–30, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).

[Lavie and Agarwal2007] Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic, June. Association for Computational Linguistics.

[Lavie and Denkowski2009] Alon Lavie and Michael Denkowski. 2009. The METEOR metric for automatic evaluation of machine translation. *Machine Translation*, 23.

[Lavie et al.2008] Alon Lavie, Alok Parlikar, and Vamshi Ambati. 2008. Syntax-driven learning of sub-sentential translation equivalents and translation rules from parsed parallel corpora. In *Proceedings of the ACL-08: HLT Second Workshop on Syntax and Structure in Statistical Translation (SSST-2)*, pages 87–95, Columbus, Ohio, June. Association for Computational Linguistics.

[LDC2005] LDC. 2005. Linguistic Data Annotation Specification: Assessment of Fluency and Adequacy in Translations. Revision 1.5.

[Levenberg et al.2010] Abby Levenberg, Chris Callison-Burch, and Miles Osborne. 2010. Stream-based translation models for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of*

the North American Chapter of the Association for Computational Linguistics, pages 394–402, Los Angeles, California, June. Association for Computational Linguistics.

[Liu et al.2006] Yang (1) Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia, July. Association for Computational Linguistics.

[Liu et al.2009] Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 558–566, Suntec, Singapore, August. Association for Computational Linguistics.

[Liu et al.2011] Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2011. Better evaluation metrics lead to better machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 375–384, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

[Lopez2008a] Adam Lopez. 2008a. Machine translation by pattern matching. In *Dissertation, University of Maryland*, March.

[Lopez2008b] Adam Lopez. 2008b. Tera-scale translation models via pattern matching. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 505–512, Manchester, UK, August. Coling 2008 Organizing Committee.

[Maarit Koponen and Specia2012] Luciana Ramos Maarit Koponen, Wilker Aziz and Lucia Specia. 2012. Post-editing time as a measure of cognitive effort . In *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, pages 11–20, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).

[Macherey et al.2008] Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 725–734, Honolulu, Hawaii, October. Association for Computational Linguistics.

[Manber and Myers1993] Udi Manber and Gene Myers. 1993. Suffix arrays: A new method for on-line string searches. 22:935–948.

[Martínez-Gómez et al.2012] Pascual Martínez-Gómez, Germán Sanchis-Trilles, and Francisco Casacuberta. 2012. Online adaptation strategies for statistical machine translation in post-editing scenarios. 45:3193–3203.

[Miller and Fellbaum2007] George Miller and Christiane Fellbaum. 2007. WordNet. http://wordnet.princeton.edu/.

[O'Brien et al.2012] Sharon O'Brien, Michel Simard, and Lucia Specia, editors. 2012. *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).

[Och and Ney2002] Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

[Och and Ney2003] Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29.

[Och and Ney2004] Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. 30.

[Och et al.1999] Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

[Och2003] Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July. Association for Computational Linguistics.

[Olive et al.2011] Joseph Olive, Caitlin Christianson, and John McCary, editors. 2011. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*. Springer.

[Ortiz-Martínez et al.2010] Daniel Ortiz-Martínez, Ismael García-Varea, and Francisco Casacuberta. 2010. Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 546–554, Los Angeles, California, June. Association for Computational Linguistics.

[Ortiz-Martínez et al.2012] Daniel Ortiz-Martínez, Germán Sanchis-Trilles, Francisco Casacuberta, Vicent Alabau, Enrique Vidal, José-Miguel Benedí, Jesús González-Rubio, Alberto Sanchis, and Jorge González. 2012. The CASMACAT project: The next generation translator's workbench. In *Proceedings of the 7th Jornadas en Tecnología del Habla and the 3rd Iberian SLTech Workshop (IberSPEECH)*, pages 326–334.

[Papineni et al.2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

[Parker et al.2011] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English Gigaword Fifth Edition, June. Linguistic Data Consortium, LDC2011T07.

[Penkale and Way2012] Sergio Penkale and Andy Way. 2012. SmartMATE: An Online End-To-End MT Post-Editing Framework. In *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, pages 51–59, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).

[Porter2001] Martin Porter. 2001. Snowball: A language for stemming algorithms. http://snowball.tartarus.org/texts/.

[Poulis and Kolovratnik2012] Alexandros Poulis and David Kolovratnik. 2012. To post-edit or not to post-edit? Estimating the benefits of MT post-editing for a European organization. In *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, pages 60–68, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).

[Przybocki2009] Mark Przybocki. 2009. Nist open machine translation 2009 evaluation. http://www.itl.nist.gov/iad/mig/tests/mt/2009/.

[Przybocki2012] Mark Przybocki. 2012. Nist open machine translation 2012 evaluation (openmt12). http://www.nist.gov/itl/iad/mig/openmt12.cfm.

[Sanchis-Trilles2012] Germán Sanchis-Trilles. 2012. Building task-oriented machine translation systems. In *Ph.D. Thesis, Universitat Politcnica de Valncia*.

[Snover et al.2006] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the. Association for Machine Translation of the Americas*, pages 223–231.

[Snover et al.2009] Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2009. Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 259–268, Athens, Greece, March. Association for Computational Linguistics.

[Soricut et al.2012] Radu Soricut, Nguyen Bach, and Ziyuan Wang. 2012. The sdl language weaver systems in the wmt12 quality estimation shared task. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 145–151, Montréal, Canada, June. Association for Computational Linguistics.

[Specia and Farzindar2010] Lucia Specia and Atefeh Farzindar. 2010. Estimating machine translation post-editing effort with HTER. In *Proceedings of the AMTA-2010 Workshop Bringing MT to the User: MT Research and the Translation Industry*, pages 33–41.

[Specia and Gimenez2010] Lucia Specia and Jesús Gimenez. 2010. Combining confidence estimation and reference-based metrics for segment level MT evaluation. In *Proceedings of the Ninth Conference of the Association for Machine Translation in the Americas*.

[Specia2011] Lucia Specia. 2011. Exploiting objective annotations for measuring translation post-editing effort. In *Proceedings of the 15th International Conference of the European Association for Machine Translation*.

[Tatsumi et al.2012] Midori Tatsumi, Takako Aikawa, Kentaro Yamamoto, and Hitoshi Isahara. 2012. How Good Is Crowd Post-Editing? Its Potential and Limitations. In *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, pages 69–77, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).

[Tatsumi2010] Midori Tatsumi. 2010. Post-editing machine translated text in a commercial setting: Observation and statistical analysis. In *Ph.D. thesis, Dublin City University*, October.

[van Rijsbergen1979] C. J. van Rijsbergen, 1979. *Information Retrieval*, chapter 7. Butterworths, London, UK, 2nd edition.

[Wang and Manning2012] Mengqiu Wang and Christopher Manning. 2012. Spede: Probabilistic edit distance metrics for mt evaluation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 76–83, Montréal, Canada, June. Association for Computational Linguistics.

[Wikipedia2013] Wikipedia. 2013. Wikipedia: The Free Encyclopedia. http://www.wikipedia.org.

[Yamada and Knight2001] Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Toulouse, France, July. Association for Computational Linguistics.

[Zaidan2009] Omar F. Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.

[Zhechev2012] Ventsislav Zhechev. 2012. Machine Translation Infrastructure and Post-editing Performance at Autodesk. In *AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*, pages 87–96, San Diego, USA, October. Association for Machine Translation in the Americas (AMTA).