Machine Translation Journal manuscript No.
(will be inserted by the editor)

Survey of Data-Selection Methods in Statistical Machine Translation

Sauleh Eetemadi · William Lewis · Kristina Toutanova · Hayder Radha

Received: date / Accepted: date

Abstract Statistical Machine Translation has seen significant improvements in quality over the past several years. The single biggest factor in this improvement has been the accumulation of ever larger stores of data. We now find ourselves, however, the victims of our own success, in that it has become increasingly difficult to train on such large sets of data, due to limitations in memory, processing power, and ultimately, speed (i.e. data-to-models takes an inordinate amount of time). Moreover, the training data has a wide quality spectrum. A variety of methods for data cleaning and data selection have been developed to address these issues. Each of these methods employs a search or filtering algorithm to select a subset of the data, given a defined set of feature functions. In this paper we provide a comparative overview of research in this area based on application scenario, feature functions and search method.

Keywords Statistical Machine Translation · Data Selection · Data Cleaning · Literature Overview

S. Eetemadi
Michigan State University, Microsoft Research
E-mail: etemadys@msu.edu

W. Lewis
Microsoft Research
E-mail: wilewis@microsoft.com

K. Toutanova
Microsoft Research
E-mail: kristout@microsoft.com

H. Radha
Michigan State University
E-mail: radha@egr.msu.edu

1 Introduction

Training data for Statistical Machine Translation (SMT) has increased significantly over the past several years and is continuing to grow even further as digital multilingual communication and information dissemination become ubiquitous. However, the data is also coming in an increasingly wider spectrum of quality, from sentence-level professional translations to crowd-sourced translations, to machine-generated outputs from free online translators. This is one of the reasons why data selection and cleaning has become a common step in the development of machine translation (MT) systems. Data-selection techniques are also relevant for low-resource languages. While training data for SMT is abundant in some language pairs, development of high quality MT systems for low-resource languages remains a challenge due to lack of training data. Using human translators to produce new training data has been a common solution to this problem (Haffari et al (2009); Ambati et al (2010); Ananthakrishnan et al (2010a); Callison-Burch and Dredze (2010)), yet the cost of translation has been a limiting factor. As a result, data-selection techniques have been developed to select monolingual data that, if translated, can improve the quality of an MT system the most. While industry leaders in the field have the infrastructure and resources to build and iterate over ever growing data sets, this is not cost-effective for small businesses and academics. Notwithstanding the presence of abundant training data for SMT, it is still desirable to select a subset of the data that enables the best translation quality in an SMT system.[1] Finally, offline mobile applications which generally have constrained memory limitations are another important application of data selection in MT.

As a result, a subfield of data cleaning and selection has formed in the field of MT, and some of the most practical methods for data cleaning and selection have appeared in system submission papers as preprocessing steps. This paper attempts to exhaustively review the literature in MT data selection and offer a comparative overview of the proposed methods in each application scenario.

All data-selection and cleaning algorithms select a subset of some original dataset. They differ in three main characteristics:

1. **Application Scenario**: The application scenario for each method (e.g. resource reduction for high or low-resource languages, quality improvement in general or domain adaptation settings) defines the type of data that is

[1] One point to acknowledge here is the extent to which the data supplied is optimally used. Ozdowska and Way (2009) provide comparative scores of systems built using different subsets of Europarl data (Koehn (2005)). As might be expected, performance is highest for language pairs which use parallel data constructed with that particular language direction in mind, i.e. a French-to-English MT system works best when the original language data was French, and subsequently translated into English, as opposed to (i) where the original language data was English, and translated into French, or (ii) where the source language was neither French nor English. This is an important point, yet remains largely ignored by the wider community; system developers tend to value larger amounts of (say) French–English parallel data rather than select segments of a data set based on what the original language was.

being selected (parallel or monolingual), and the problem that the method aims to solve (e.g. maximize translation quality with minimal training time memory consumption).

2. **Scoring Functions**: Since it is infeasible to train an MT system on each subset of the data and evaluate the quality of the resulting system for each subset, researchers have developed scoring functions (features) defined on subsets of the data, which are expected to correlate with the usefulness of the data subset for training a high-quality MT system. Different application scenarios motivate different scoring functions.
3. **Selection Algorithms**: Given one or more scoring functions for evaluating a subset of data, we need to find the subset that maximizes this scoring function subject to some constraints. Depending on the properties of the scoring function, exact or approximate search algorithms have been developed.

In the rest of the paper we first review the variations in the literature based on the main components listed in Fig. 1. Next we provide a comparative overview of literature as to how each work has combined the components above and how they compare with other research. In Section 2 we break out and detail the application scenarios foundational to data-selection algorithms. In Section 3, we provide and describe the notation used in the rest of the paper. In Section 4, we formalize the issue of data selection as a constrained optimization problem and elaborate on different formulations of such problems corresponding to different application scenarios. An exhaustive list of data-selection scoring functions is presented in Section 5 followed by feature combination and parameter tuning in Section 6. The scoring functions are organized based on the statistical model (e.g. language model, translation model, . . .) they use. Search methods used to solve optimization problems are listed in Sections 7, 8 and 9. Methods of evaluating the effectiveness of data-selection methods are discussed in Section 10. We provide a comparative summary of all prior work in Section 11, along with recommendations for applying these methods, and list related areas of research in Section 12.

2 Application Scenarios

Data selection is performed in a variety of scenarios. There are two broad application scenario categories: those where the goal is to minimize resource consumption, and those focused on quality improvements or noise reduction. These in turn can be broken down into sub-categories, as described below, all based on selection criteria and scoring functions, which will be discussed in more detail in Section 4:

1. **Satisfying Resource Constraints:** A common scenario in data selection for MT is that training data size needs to be reduced due to some resource constraints (Gangadharaiah et al (2009); Chao and Li (2011a); Lewis and Eetemadi (2013)). Application scenarios in this category differ based on

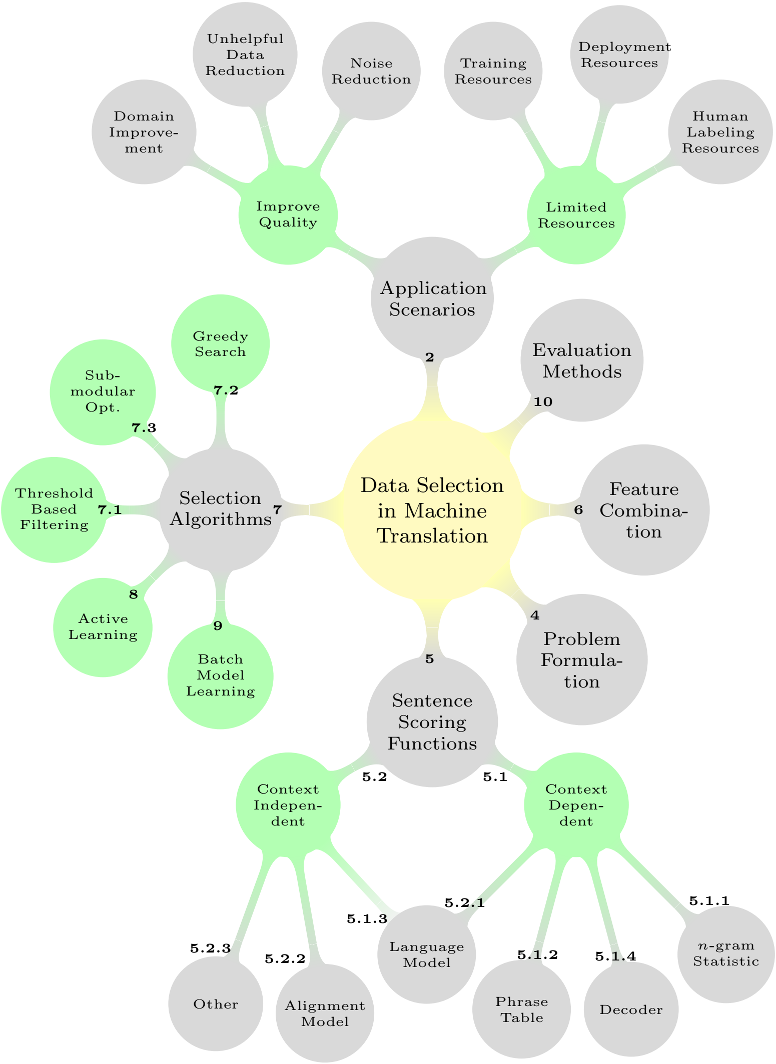

Fig. 1: Taxonomy of Data selection in MT. Numbers indicate related section numbers in this paper.

the constrained resource. Although model pruning can be used to satisfy deployment size constraints (Goodman and Gao (2000)), selecting a subset of the training data can be used to satisfy any of the resource constraint listed below. With an exponential number of subsets to choose from, the goal is to choose a subset of the training data that will ultimately result in the highest translation quality.

(a) **Training Resources:** With the increasing availability of training data for SMT, training-resource requirements (e.g. number of computers, main memory size or training time) increase as well, although Dyer et al (2008) present a scalable framework for some alignment and phrase extraction algorithms. This framework enables training on larger data sets, but the underlying tradeoff between training resources and quality of MT output remains. Hence, the goal here is to reduce training resource requirements by training on a subset of the training data with little or no impact on translation quality.

   (b) **Deployment Size:** In scenarios where a translation system is hosted on devices with limited hardware capabilities, such as mobile devices, it is desirable to produce translation models that are small in size and can be hosted on such devices with minimum impact on translation quality. One method for achieving this objective is by selecting a subset of the training data (Eck et al (2005)). Alternatively, this objective can be achieved by pruning the translation models (Such methods are outside the scope of this paper. See Zens et al (2012) for a comparison of different pruning methods).

   (c) **Manual Translation Cost:** When improving translation quality for low-resource languages[2] where pre-existing parallel training data is limited, a common approach is to select a subset of a monolingual corpus to be manually translated by humans and added to the training data (Ambati et al (2010); Ananthakrishnan et al (2010a)). Since there is a cost element involved, the goal is to achieve the highest improvement at a fixed cost or meet a predetermined quality bar with minimum cost.

2. **Quality Improvement:** Unlike the resource constraint scenario, the focus of applications that follow this scenario is on quality improvement. In these scenarios, the goal is to select a data subset which will result in a higher quality SMT system compared to an SMT system trained on the full data set. This can be done by filtering out sentences that are noisy or that have a difficult-to-learn translation phenomenon (e.g. phrase-based translation systems (Koehn et al (2003)) often learn incorrect phrase translations from freely translated[3] text (Han et al (2009))), or selecting a subset that is relevant to a domain different from the domain of the input data. Although data size will often be reduced in these scenarios, that is a side-effect, not the goal.

   (a) **Noise reduction:** A common source of training data is from crawled multilingual web sites (Resnik (1999)). Web data can be very noisy and although robustness of SMT systems to noise has been studied (Goutte et al (2012); Pecina et al (2014)), noise reduction remains a common preprocessing step for this kind of data (Denkowski et al (2012)). For parallel data, a pair of sentences that are not good translations of each other can be considered noise.

[2] Low-resource languages are languages that have relatively little monolingual or parallel data available for training, tuning and evaluation.

[3] Free translation or freely translated text – contrary to literal, direct or word-for-word translation – conveys the overall meaning of a sentence or phrase without necessarily a word-for-word correspondence between source and translated text (Han et al (2009)).

(b) **Reduction of Unhelpful Data:** Although the objective is similar and also partially achieved by noise reduction, it may be desirable to filter out data that are not traditionally classified as noise. For example, non-literal translations are not desirable for training phrase-based SMT systems (at least using current technology).

(c) **Domain Improvement:** A common approach for improving translation quality is to train domain-specific translation models (Moore and Lewis (2010); Axelrod et al (2011); Banerjee et al (2011)). To that end, a subset of the training data that is more representative of the target domain can be selected. In this task, an in-domain data set is used to guide the training data-selection process.

3 Notation and Terminology

Prior work in data selection in MT has borrowed terminology and notation from several related fields including active learning, machine learning and quality estimation. We list the terms used in the literature for each concept in data selection and present the terms and notations we use throughout the paper.

Data selection in MT has also been referred to as data cleaning (Okita (2009); Jiang et al (2010)), noise reduction (Khadivi and Ney (2005); Okita et al (2009); Taghipour et al (2010)), improving training data quality (Liu et al (2010b); Adesam (2012)) and active learning (Ambati et al (2010); Ananthakrishnan et al (2010b)). Although we use all of these terms in the context of corresponding scenarios or methods, we refer to the general task of selecting a subset of data for training as "data selection" independent of its purpose or method.

The data-selection task is to select the 'best' subset of the data for MT model training. We refer to the full set of available data as "**selection pool**" or S_{pool} and denote the 'best' subset by **"optimum subset"**, S^*, regardless of the optimization criteria. The unit of data is always sentences or sentence pairs.[4] We do not specifically distinguish parallel data or sentence pairs versus monolingual data or sentences when it is apparent from the context. When a distinction between two sides of a parallel corpus is necessary, we use the letter "s" with appropriate casing to indicate the source side of the parallel data and **"t"** for the target side. The selection pool is also called "unlabeled data" in some prior work (Haffari (2009)). In some scenarios there is a seed parallel or monolingual corpus given as part of the data-selection task. We call this data set the "**seed corpus**" and denote it by S_{seed}. In this scenario, the assumption is that the seed corpus is already selected and the task is to select a subset of the selection pool to be added to the seed corpus. Data-selection tasks for domain adaptation are also given an "**in-domain**" development set, S_{in} and an "**out-of-domain**" development set, S_{out}. In some cases, the "selection pool" is also used as the out-of-domain development set. A single sentence in any of

[4] The terms "translation units" or "utterances" could also be used, but they are functionally equivalent in this context.

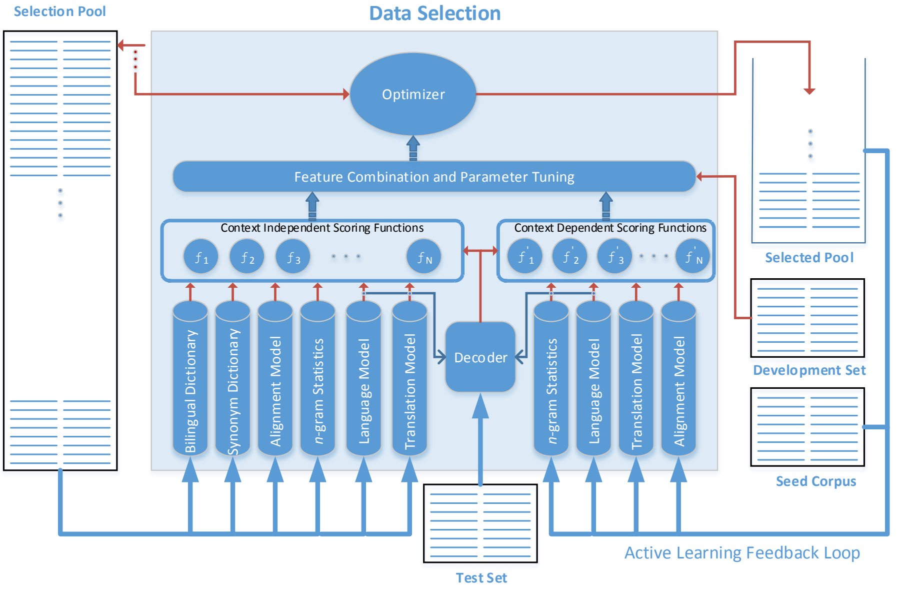

Fig. 2: Data Selection: This diagram demonstrates the role of each data set in the data-selection process.

the sets defined above is lowercase s_i to denote the i^{th} sentence in the data set. In most proposed methods for data selection, one or more functions are used to evaluate the usefulness of a sentence or sentence pair. The literature calls such functions: features, feature functions, objective functions and "**scoring functions**". We choose the last term and denote it as $f(s_i)$. Many methods proposed for data selection use iterative selection algorithms where one or more sentences from the selection pool are selected in each iteration. While the algorithm has selected j sentences, the sentences that have been selected so far are called the **"selected pool"** and are denoted as S_1^j (sentences 1 through j). In this context it is assumed that after each iteration, the "selected pool" is added to the "seed corpus". The remaining sentences are noted as "**candidate pool**", $S_{candidate}$. Scoring functions that depend on the selected pool are called "**context-dependent**" scoring functions and are denoted as $f(s_{j+1}|S_1^j)$.

Translation of sentence s_i using models trained on data set S is denoted as $t_i = T_{\mathcal{X}(S)}(s_i)$ where $\mathcal{X}(S)$ is any model trained on data set S (e.g. language model, $\mathcal{LM}(S)$, word alignment model, $\mathcal{AM}(S)$, or translation model, $\mathcal{TM}(S)$). Lowercase s_1^m indicates words 1 through m in sentence s. $P_{\mathcal{LM}(S)}(s_i)$ represents the probability of sentence s_i appearing in the evaluation set according to a language model trained on corpus S. $P_{\mathcal{TM}(S)}(s_i, t_i)$ and $P_{\mathcal{AM}(S)}(s_i, t_i)$ are interpreted similarly for a **translation model** and a **word alignment model**.

Figure 2 demonstrates the role of each data set in the data-selection process. The selection pool can be used to build probabilistic models (e.g. language model, translation model) to provide statistical insight. These models are computed once and not updated. In contrast, statistical models computed using the selected pool separately or combined with the seed corpus are updated as the selected pool changes during the data-selection process. Scoring functions that depend on the models that need to be updated are called “context-dependent scoring functions”. In addition, a decoder using models built by the selection or selected pool can be used as another input to a context-dependent or context-independent scoring function. A development set can be used to train weights when more than one feature function is used. Ultimately, the final scoring function is used to score sentences or sentence pairs in the selection pool with an optimizer used to select the optimum subset of the selection pool. The optimization step is often done by greedily adding sentence pairs from the selection pool into the selected pool. For context-dependent functions, the models or statistics need to be recomputed after the selection of each sentence pair, or N sentence pairs in the case of batch learning.

4 Problem Formulation

Data selection can be formulated as a constrained optimization problem. The constraint is that the optimum subset has a given maximum size, which is usually measured by the number of sentences or sentence pairs in the subset, denoted by $|S|$.[5] The true scoring function that a data-selection method aims to maximize measures the expected translation quality using data drawn from some desired target distribution (which could be of general domain or a specific target domain in the case of domain adaptation).

Below we formulate a constrained optimization problem in a generic form, which expresses what data-selection methods aim to achieve in theory. Since it is impractical to solve this constrained optimization problem,[6] different data-selection methods use alternative formulations. The true scoring function is to select a subset, S, from the selection pool, S_{pool}, that maximizes the expected translation quality[7] when translated using models trained on S for any data sample, D, that is drawn from some desired target distribution $\mathrm{P}(n\text{-gram})$, subject to some size or resource constraint C, as shown in Equation (1):

$$S^* = \underset{S \subset S_{pool}, \mathrm{Size}(S) < C}{\arg\max} \mathbb{E}\left[\mathrm{BLEU}\left(D_{ref}, T_{X(S)}\left(D_{src}\right)\right)\right] \tag{1}$$

D_{src} : Source side of D

D_{ref} : Reference translation for D_{src}

[5] Size can also be measured in total number of characters, tokens, token types, etc depending on the scenario.

[6] The number of subsets of size m from a set of size n is $\frac{n!}{m! \times (n-m)!}$. Since this value is exponential in n, to be precise: $\frac{n}{k}$, it is impractical to do an exhaustive search enumerating all subsets.

[7] We will use BLEU as the automated metric here, but other measures are certainly viable. **Bil**ingual **E**valuation **U**nderstudy is the most popular automatic evaluation metric based on n-gram matches between translation output and reference translations (Papineni et al (2002); Koehn (2009))

The value for $\mathbb{E}\left[\text{BLEU}\left(D_{ref}, T_{X(S)}\left(D_{src}\right)\right)\right]$, is usually estimated using a blind test set, S_{Test}, drawn from the desired target distribution $P(n\text{-gram})$, as in Equation (2):

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \ \text{BLEU}\left(S_{Test_{ref}}, T_{X(S)}\left(S_{Test_{src}}\right)\right) \tag{2}$$

Consequently, the task of data selection is to find the best subset of sentences that will result in the highest translation quality as approximated on a blind test set which is not available during training. Some application scenarios provide a constraint on the size of the subset. There are, therefore, two technical problems to solve:

1. Efficient estimation of the quality of an MT system trained on a set S using an unseen test set.
2. Finding the data subset S^* that maximizes this estimated quality.

A variety of models have been defined for the usefulness of a sentence pair. They either use

- a single scoring function on data subset S, or
- a model which combines multiple scoring functions, or
- in the case of active learning, an SMT system (or some elements of it, such as a language or translation model) which is trained on the training data and evaluated against a blind test set.

An example of a scoring function is n-gram[8] coverage of the data subset with respect to the selection pool. Thus a subset that covers a larger portion of all n-grams is assumed to result in a higher quality MT system. Multiple scoring functions can also be combined using a linear model for estimation. Equation (3) shows a formulation of the optimization problem from Equation (2) where a combination of scoring functions is used to model translation quality:

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{i=1}^{\substack{\textit{Feature}\\ \textit{Count}}} \lambda_i \ \mathcal{F}_i(S) \tag{3}$$

$$\mathcal{F}_i(S) : \text{corpus-level scoring function}$$

$$\lambda_i : \text{feature weight}$$

Once a model for scoring the goodness of data subsets has been defined, the remaining problem is searching for the best subset of a given maximum size. Depending on the scoring functions, the search problem can be extremely hard and necessitate approximate search techniques. For practical purposes, the scoring functions over data subsets can be decomposed into increments when adding each sentence one by one in a given order. Given this decomposition,

[8] In this paper, unless otherwise specified, word n-grams are referred to as simply n-grams for brevity. A word n-gram is a sequence of n words that appear in natural language text.

a corpus subset can be selected incrementally by adding individual sentences. Equation (4) shows such a decomposition:

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{j=1}^{|S|} \sum_{i=1}^{\substack{\mathit{Feature}\\ \mathit{Count}}} \lambda_i \, f_i(s_j | S_1^{j-1}) \tag{4}$$

$$f_i(s_j | S_1^{j-1}) : \text{context-dependent sentence-level scoring function}$$

Equation (4) assumes conditional independence for the scoring function between sentences in the selection pool given the selected pool. In contrast, Equation (5) assumes complete independence. If the sentence-level scores are independent of previously selected sentences (e.g. sentence length), a simple threshold-based filtering algorithm will provide the optimum solution (Kirchhoff and Bilmes (2014)) to the constraint optimization problem with computational complexity of $\mathcal{O}(|S| \times \log |S|)$ for sorting sentences based on their scoring function value and $\mathcal{O}(|S|)$ for satisfying the resource constraint. This is often the case for scoring functions motivated by data cleaning and noise reduction, e.g. the source-to-target sentence length ratio. Sentence-level scoring functions that are motivated by minimizing resource constraints are often dependent on previously selected sentences, e.g. number of new n-grams in a sentence.

$$S^* = \underset{S \subset S_{pool}, \text{Size}(S) < C}{\arg\max} \sum_{s \in S} \sum_{i=1}^{\substack{\mathit{Feature}\\ \mathit{Count}}} \lambda_i \, f_i(s) \tag{5}$$

$$f_i(s) : \text{sentence-level scoring function}$$

In cases such as (5), an incremental search algorithm such as greedy search is often used to solve the constrained optimization problem. Selection algorithms are discussed in more detail in Section 7.

5 Sentence-Level Scoring Functions

As explained in Section 4, the constrained optimization problem is often broken down using a combination of sentence-level scoring functions to enable an efficient solution to the optimization problem. These functions can be categorized as follows.

- **Context-Independent Functions** which depend on nothing but the candidate sentences in question.
- **Context-Dependent Functions** which depend on the selected pool.

Threshold filtering can only be used for context-independent functions. For context-dependent functions, a greedy incremental search algorithm is often used to solve the optimization problem.

5.1 Context-Dependent Functions

The values or scores of context-dependent functions depend on statistics from the selected data, in addition to the sentence being scored. In a typical data-selection algorithm, all sentences are scored and one or more[9] sentences with the highest score are added to the selected pool. This requires the statistics in use to be recomputed in addition to all candidate sentence scores. We categorize the scoring functions in this group based on the statistics or models that need to be calculated over the selected sentences in order to assign scores to candidate sentences.

5.1.1 N-Gram Coverage Functions

Functions in this group are motivated by finding a subset of the selection pool that best represents the entire set using n-gram statistics. In other words, when selecting a subset of training data, it is often desirable to preserve the vocabulary and its context (n-grams). The simplest way of using n-gram statistics for this purpose is to ensure all n-grams of up to a certain length are present in the selected subset. For this purpose, it is sufficient to keep the n-gram count for the selected pool. For each candidate sentence, the score is equal to the number of new n-grams it contains where the n-gram length is less than or equal to N.

$$f_1(s_j|S_1^{j-1}) = \frac{1}{norm(s_j)} \sum_{n=1}^{N} \sum_{ng \in \mathrm{NG}(s_j,n)} weight(ng, j-1) \tag{6}$$

$$\begin{aligned} \mathrm{NG}(s,n) &: \text{ All } n\text{-grams of size } n \text{ in sentence } s \\ weight(ng,m) &: \text{ Weight of } n\text{-gram } ng \text{ given selected pool } S_1^m \text{ are selected} \\ norm(s) &: \text{ Normalization factor for sentence } s \end{aligned}$$

Equation (6) covers much of the prior work depending on the choice of $weight(ng,m)$ and $norm(s)$. Eck et al (2005) present several scoring functions with $weight(ng)$ equal to 1. In their simplest scoring function they only count the number of new n-grams in a sentence and normalize by the length of the sentence, as in Equation (7):

$$\begin{aligned} weight_1(ng,m) &= \begin{cases} 0 & ng \in \bigcup_{i=1}^{m} \mathrm{NG}(s_m,n) \\ 1 & \text{otherwise} \end{cases} \\ norm_1(s) &= |\,\mathrm{NG}(s,n)| \end{aligned} \tag{7}$$

[9] Batch-learning techniques are often used to make data-selection methods practical. See Section 9 for details.

In this approach rare and frequent n-grams are valued equally. In their effort to assign a higher weight to more frequent n-grams in the selection pool, S_{pool}, Eck et al use the n-gram frequency as the weight in their improved scoring function (Equation (8)). Using normalization factors of $|s|^i$ where i takes on values of 0, 1 and 2, are other variations attempted by this work.

$$\begin{aligned} weight_2(ng, m) &= \begin{cases} 0 & ng \in \bigcup_{i=1}^{m} \mathrm{NG}(s_m, n) \\ \mathrm{Freq}(ng, S_{pool}) & \text{otherwise} \end{cases} \qquad (8) \\ \mathrm{Freq}(ng, S) &= \sum_{s \in S} \sum_{ng \in \mathrm{NG}(s, len(ng))} 1 \end{aligned}$$

A clear shortcoming of the function in (8) is that once an n-gram exists in the selected sentences, it is of zero value when selecting new sentences. This does not allow for the scoring function to discriminate against an n-gram with frequent occurrences in the selected sentences versus an n-gram that has only appeared once. This is the motivation for introducing a feature decay function for the n-gram weight (Ambati (2011); Biçici and Yuret (2011)).

$$\begin{aligned} weight_3(ng, m) &= \mathrm{Freq}(ng, S) * \mathrm{e}^{-\lambda * \mathrm{Freq}(ng, \{s_1 \ldots s_m\})} \qquad (9) \\ &\lambda : \text{Exponential decay hyper parameter} \end{aligned}$$

For their last variation on the weight function, Eck et al (2005) use the cosine similarity between TF-IDF[10] vectors of the selected pool and sentences in the candidate pool. In an information-retrieval setting for TF-IDF, all sentences in the selected pool play the role of the search query while each sentence in the candidate pool is considered a potential matching document. The goal in this case is to find the document (candidate sentence) that is most dissimilar to the search query (selected pool). Defining the weight function, $weight(ng, m)$, as follows, results in Equation (6) which uses cosine similarity of TF-IDF vectors as defined above, as the sentence-scoring function.

$$\begin{aligned} tf(ng, S) &= \mathrm{Freq}(ng, S) \\ idf(ng, S) &= \frac{|S|}{\sum_{s_i \in S} \mathbb{1}\,(ng \in \mathrm{NG}(s_i, len(ng)))} \\ \mathbb{1}\,(ng \in \mathrm{NG}) &= \begin{cases} 1 & ng \in \mathrm{NG} \\ 0 & \text{otherwise} \end{cases} \\ weight_4(ng, m) &= tf(ng, S_1^m) * \mathrm{Log}\, idf(ng, S_{m+1}^{|S|}) \qquad (10) \end{aligned}$$

Since the goal of the TF-IDF weight function is to find the most dissimilar sentence, the overall optimization problem (Equation (5)) turns into a minimization, unlike other n-gram-based scoring functions.

[10] Term Frequency - Inverse Document Frequency

		Selection Pool		
	Phrase Pairs	high frequency	low frequency	zero frequency
Seed Corpus	high frequency			
	low frequency	B^*	D^*	
	zero frequency	A^*	C^*	

Table 1: Exploration versus Exploitation. A^*: Improved coverage for important phrase pairs;[11] B^*: Improved estimation for important phrase pairs; C^*: Improved coverage for phrase pairs; D^*: Improved estimation for phrase pairs.

5.1.2 Phrase-Table-Based Functions

Scoring functions in this section require a phrase table. Depending on the function, the "seed corpus", the "selection pool", the "selected pool" or "candidate pool" or a combination thereof are used to train the phrase table. When selecting from a monolingual pool of sentences, a full SMT system trained on the seed corpus can be used to turn the monolingual corpus into a parallel corpus to train the phrase table (Haffari (2009)). Haffari (2009) introduces the idea of *exploration* versus *exploitation* in this context. The idea is to 'explore' by selecting sentences with new phrase pairs to expand coverage while 'exploiting' phrase pairs that are not new, to improve their probability estimation. We use Table 1 to further explain this idea. Phrase pairs that occur frequently in the selection pool but do not occur (or occur rarely) in the seed corpus are of most value. If they do not exist in the seed corpus (A^*), they add new coverage (exploration). On the other hand, they provide better estimation if they occur rarely in the seed corpus (B^*, exploitation). Phrase pairs with low frequency in the selection pool can also be useful if they occur rarely in the seed corpus (D^*) or do not occur at all (C^*). Although this intuition is explained in the context of phrase-pairs, it is applicable to n-gram- and language model-based scoring functions as well.

Haffari introduces a phrase-pair utility function to capture exploration versus exploitation (Equation (11)). The sentence-scoring function is then computed using arithmetic (Equation (12)) or geometric (Equation (13)) average of the phrase pair utility function over all phrase pairs.

[11] The assumption here is that a phrase pair that occurs frequently in the selection pool is likely to be used in a translation task, and thus is important.

$$\text{PPUtil}(pp|(s,t)_{1}^{j-1}) = \frac{P(PP_1 = pp|PP_1 \in \bigcup_{k=1}^{j-1} \text{PPs}(s_k, t_k))}{P(PP_2 = pp|PP_2 \in \bigcup_{k=j}^{|S|} \text{PPs}(s_k, t_k))} \tag{11}$$

pp : A phrase pair mapping a phrase in one language
to its translation in a second language.

$\text{PPs}(s,t)$: The set of all phrase pairs extracted
from sentence pair (s,t).

$$f_2((s_j,t_j)|(s,t)_1^{j-1}) = \left(\prod_{pp \in \text{PPs}((s_j,t_j))} \text{Log}\,\text{PPUtil}(pp|(s,t)_{1}^{j-1}) \right)^{\frac{1}{|S|}} \tag{12}$$

$$f_3((s_j,t_j)|(s,t)_1^{j-1}) = \frac{1}{|S|} \sum_{pp \in \text{PPs}((s_j,t_j))} \text{PPUtil}(pp|(s,t)_{1}^{j-1}) \tag{13}$$

Ambati (2011) takes a different approach in using a phrase table for scoring sentences. The goal is to give a higher score to source phrases where the phrase table is more uncertain about their translation. Since entropy is a measure of uncertainty, Ambati defines phrasal entropy to calculate the level of uncertainty a phrase table has regarding a source phrase, as in Equation (14):

$$\text{PhrEnt}(p_s|(s,t)_1^j) = \frac{\sum_{p_t \in \text{Trans}(p_s)} -P_{TM(S_1^j)}(p_t|p_s) * \log P_{TM(S_1^j)}(p_t|p_s)}{|\,\text{Trans}(p_s)|}$$

$\text{Trans}(p_s)$: Target phrases for phrase p_s according to
phrase table trained on $(s,t)_1^j$ (14)

Using phrasal entropy,[12] the scoring function for a sentence is defined as the sum of the phrasal entropy over all source phrases in a sentence. This scoring function can be extended to use two different phrase tables (one trained from source-to-target language and the other target-to-source). Using the second phrase table, uncertainty about target phrases can also be taken into account.

5.1.3 Language Model-Based Functions

Language model-based scoring functions are similar to n-gram-based scoring functions. In both cases, n-grams are the basis for scoring sentences. The advantage of language model-based scoring functions is their higher accuracy in

[12] By definition the conditional entropy of $H(Y|X)$ is defined as $\sum_{x \in X, y \in Y} p(x,y) \log p(y|x)$. However, the definition provided for PhrEnt here is missing a $P_{TM(S_1^j)}(p_s)$ inside the summation but outside the log. We recognize this inaccuracy but keep formula (14) consistent with the referenced work (Ambati (2011)).

estimating n-gram probability, especially when the data is sparse, due to backoff and smoothing strategies used in language modeling. In contrast, training a language model has a higher computational complexity compared to collecting n-gram statistics for n-gram-based scoring functions.

In the absence of a phrase table (when parallel data is not available or less computational intensity is desirable), Haffari et al suggest using a language model over all possible phrases (all n-grams of up to a certain size) to calculate the equivalent of the phrase-pair utility function defined in Equation (11), n-gram utility. N-gram utility follows the same principle, but uses n-grams and two language models – one trained on the seed corpus and another trained on the selection pool – instead of phrase pairs and two phrase tables, as in Equation (15):

$$\text{NGramUtility}(ng|(s,t)_{\mathit{1}}^{\mathit{j-1}}) = \frac{P_{\mathcal{LM}(S_j^{|S|})}(NG_1 = ng)}{P_{\mathcal{LM}(S_1^{j-1})}(NG_2 = ng)} \tag{15}$$

$$P_{\mathcal{LM}(S_1^{j-1})}(NG = ng) : \text{Probability of } n\text{-gram } ng \text{ according to}$$
$$\text{language model trained on } S_1^{j-1}.$$

The sentence-scoring function is then defined as an arithmetic or geometric average of the n-gram utility function of all n-grams up to a certain length derived from the sentence. In a slight variation to Equation (15), Mandal et al (2008) use perplexity[13] instead of probability using the same language models.

Liu et al (2010a) attempt to assign the highest score to sentences with most informative n-grams. They measure informativeness of n-grams by n-gram[14] entropy. In addition, they multiply the entropy by the square root of the n-gram length to encourage longer phrases as they suggest longer phrases contribute to better translation, as in Equation (16):

$$\begin{aligned}\text{NGEntropy}(ng|S_{\mathit{1}}^{\mathit{j-1}}) = -\log\Big(P_{\mathcal{LM}(S_j^{|S|})}(ng)\Big) * \sqrt{len(ng)}\\ * \mathbb{1}\left(ng \in \text{NG}(S_j^{|S|})\right)\end{aligned} \tag{16}$$

See Equation (6) for the definition of $\text{NG}(S_j^{|S|})$. Equation (16) uses individual entropy as a measure of informativeness. We would like to point out that a more appropriate use of entropy would be using it as a measure of uncertainty. For this purpose, given an n-gram ng of size m, $\{w_1 \ldots w_m\}$, the entropy of the n-gram can be defined in the context of all n-grams of size $m{+}1$, $\{w_1 \ldots w_{m+1}\}$ that have n-gram ng as their prefix, as in Equation (17):

[13] The perplexity of a random variable is defined as two (or any given base number) to the power of its entropy. In natural language processing this function is commonly used as a measure of how surprised a language model is when observing a sequence of words.

[14] For all practical purposes in their work (Liu et al (2010a)), a phrase is the same as an n-gram as they consider phrases for a sentence to be all n-grams of up to a certain length.

$$\underset{\mathcal{LM}(S_1^j)}{\text{NGEntropy}}(w_1^m) = -\sum_{w \in \text{Vocab}(S_1^j)} P_{\mathcal{LM}(S_1^j)}(w|w_1^m) * \log\left(P_{\mathcal{LM}(S_1^j)}(w|w_1^m)\right)$$
$$\text{Vocab}(S_1^j) : \text{The set of all unique words in } S_1^j. \quad (17)$$

Finally, Ambati attempts to make the probability distribution function of the final selected pool as close as possible to the probability distribution function of the selection pool. Two language models are trained for this purpose. The first language model is trained on the seed corpus and selected pool, while the second language model is trained on the selection pool. The aim is to minimize the Kullback-Leibler divergence (Kullback and Leibler (1951))[15] between these two probability distribution functions. To achieve this, the scoring function for a sentence equals the sum of its n-gram contributions to to the overall KL divergence, as in Equation (18):

$$\text{KL-Div}(ng) = P_{\mathcal{LM}(S_1^{j-1})}(ng) * \log \frac{P_{\mathcal{LM}(S_1^{j-1})}(ng)}{P_{\mathcal{LM}(S_1^{|S|})}(ng)} \quad (18)$$

Based on Equation (18), the sentence-level scoring function is the sum of the KL-Div function over all possible n-grams in a sentence.

5.1.4 Decoder-Based Functions

Decoder-based scoring functions require a full SMT system to be trained over the seed corpus, the selection pool or both. If the goal is to select from a parallel pool of sentences, the simple approach is to use a full SMT system trained over the seed corpus to translate sentences in the selection pool. The scoring function for a sentence is based on the error or the distance between the produced translation and the target side of the sentence pair. The idea is that if the SMT system makes an error on a sentence, the sentence is likely to contain useful information. Most of the functions introduced in this category assume the selection pool is monolingual and so use other means to measure the likelihood that the decoder will produce an erroneous translation.

Haffari uses an SMT trained on the seed corpus to translate the monolingual selection pool. The scoring function is obtained by normalizing the model score produced by the decoder using sentence length. One drawback of this scoring function is that, although decoder score is an objective score when comparing translations of the same source sentence, it is not a comparable score when considering different source sentences. However, significant differences in this score can be a meaningful measure and can effectively be used for data selection (Haffari et al (2009); Ambati (2011)).

[15] Kullback-Leibler divergence is an information-theoretic measure of the distance between two probability distributions.

Another approach to measuring translation error is using **Round Trip Translation Accuracy** ((Somers (2005); Haffari et al (2009))). In this method two SMT systems are trained on the seed corpus, one in each direction. For each sentence in the selection pool, it is translated once from source to target and then from target to source. The scoring function is then the error function between the final translation and the original sentence.

Inter-System Disagreement is another approach developed by Mandal et al (2008). In this method different SMT systems (e.g. a hierarchical system (Chiang (2005)) as well as a phrase-based system) are trained on a seed corpus. Next, sentences in a parallel development set are translated using all statistical translation systems. For each translation system, an inter-system disagreement error is calculated for its translation output using all other systems' output as references. The target side of the development set is then used to train a linear regression model that predicts the final translation error based on the inter-system disagreement error from each translation system. Once the linear regression model is learned, inter-system disagreement errors are calculated for the selection pool and fed into the linear regression model to produce the final scoring function. The idea is that the sentences where the translation output from different systems differ the most are the most poorly translated sentences and thus the most informative if added to the training data together with an appropriate human translation.

The work of Ananthakrishnan et al differs from all other work in this category, since it does not use a sentence-level scoring function (Ananthakrishnan et al (2010a)). Instead, they train a sentence-level pairwise comparator classifier to sort sentences in the selection pool based on their estimated translation error reduction. The pairwise comparator uses *n*-gram features to compare sentences. However, we include this work in this section, since a decoder with models trained on the seed corpus is used to create the development set. First, a parallel corpus is required to create the development set. Next, the source side of the parallel corpus is decoded and the TER[16] is calculated for each sentence pair. Ananthakrishnan et al suggest if the sentence with the highest TER is added to the seed corpus, it will likely reduce the translation error the most. Sorting the sentence pairs in the development set according to their TER score produces the final development set. They train the pairwise comparator weights to best match the order of sentences in the development set. After the comparator is trained, it is used to sort the sentences. The scoring function in this case is the rank of each sentence in the sorted list.

Finally, Kauchak (2006) attempts to estimate the contribution of each sentence pair in the selection pool by sampling a large number of random subsets of the selection pool with replacement and training an SMT system over each subset of the training data. After calculating the performance of each system (using BLEU score, say), a contribution score is estimated for each sentence pair based on its membership in different subsets and the performance

[16] Translation Edit Rate (TER) measures the amount of editing a human would need to perform on an MT output to exactly match a reference translation (Snover et al (2006)).

of those subsets. This method is very computationally intensive and its use is not feasible in practical scenarios. However, it can be calculated on small data sets and used as an oracle score to evaluate other data-selection features or tune hyper-parameters for various data-selection methods.

5.2 Context-Independent Functions

Context-independent functions have mostly been inspired by features used in the area of Translation Quality Estimation (Ueffing and Ney (2007)), where, given a sentence and its translation, the quality of the translation is estimated in the absence of a reference translation. We have grouped these functions based on the statistic or model that is required for their computation.

5.2.1 Language Model-Based Functions

Language models can be used in different ways to assess the usefulness of a sentence or sentence pair. Denkowski et al (2012) use an external source and target language model trained on clean data to filter out ungrammatical sentences (Equation (19)). In this case the scoring function is the length-normalized language model score of the sentence. Allauzen et al (2011) use the language-model perplexity score of the sentence as its scoring function to filter out foreign-language sentences (Equation (20)).

$$
\begin{aligned}
\mathcal{f}_4(s_i) &= P_{\mathcal{LM}(S_{\text{clean}})}(s_i) && (19)\\
\mathcal{f}_5(s_i|S_{\text{clean}}) &= 2^{H^{ind}_{\mathcal{LM}(S_{\text{clean}})}(s_i)} && (20)\\
H^{ind}_{\mathcal{LM}(S)}(s_i) &= -P_{\mathcal{LM}(S)}(s_i) * \log P_{\mathcal{LM}(S)}(s_i) \qquad (\textit{individual entropy})
\end{aligned}
$$

Yasuda et al (2008) use a language model trained on an in-domain[17] development set to score sentences in the selection pool. In this approach the scoring function equals the geometrical average of the source and target entropy[18] of the sentence pair according to the in-domain language models, as in Equation (19):

$$
\mathcal{f}_6(s_i|S_{\text{in}}) = \sqrt{H^{ind}_{\mathcal{LM}(S_{\text{in-src}})}(s_i) * H^{ind}_{\mathcal{LM}(S_{\text{in-tgt}})}(s_i)} \qquad (21)
$$

Since the language model used in computing the entropy is an estimate of the true probability distribution function, cross-entropy[19] is used instead. A

[17] The seed corpus can be used for this purpose as well.

[18] Entropy, $H(X)$, measures the amount of information or uncertainty in a random variable (Manning and Schütze (1999)).

[19] The cross-entropy of a random variable, X, with the true probability distribution function of $P(X)$ and an estimated probability distribution function of $Q(X)$ is formally defined as the sum of the entropy of X plus the KL-divergence between $P(X)$ and $Q(X)$: $H(P(X), Q(X)) = H(X, P(X)) + D_{KL}(P(X)||Q(X))$ (Cover and Thomas (1991)).

uniform probability distribution function is used to estimate cross-entropy. The goal in this method is to give sentences similar to the in-domain development set a higher score. However, the shortcoming of this method is that it will also give generally popular sentences (that are also popular in the in-domain development set) a high score as well. This is not desirable in a domain-adaptation data-selection task. Moore and Lewis address this issue by introducing a second language model trained on a general-domain development set (Moore and Lewis (2010)). In their approach, they use the **cross-entropy difference** of the two language models to score sentences, as in Equation (22):

$$f_7(s_i|S_{\text{in}}, S_{\text{out}}) = H^{ind}_{\mathcal{LM}(S_{\text{in}})}(s_i) - H^{ind}_{\mathcal{LM}(S_{\text{out}})}(s_i) \tag{22}$$

Axelrod et al (2011) extend the cross-entropy difference method to include the target side of parallel data. In their method (**bilingual cross-entropy difference**), two separate in-domain and general-domain language models are trained for the source and target language. For a given sentence pair, its score is calculated by adding together the cross-entropy difference of source and target, as in Equation (23):

$$f_8(s_i|S_{\text{in}}, S_{\text{out}}) = f_7(s_{isrc}|S_{\text{in-src}}, S_{\text{out-src}}) + f_7(s_{itgt}|S_{\text{in-tgt}}, S_{\text{out-tgt}}) \tag{23}$$

5.2.2 Alignment Model-Based Functions

Alignment-based scoring functions[20] are only applicable to parallel training data selection. These scoring functions attempt to quantify the translation quality and accuracy of the sentence pair. Any word alignment model (e.g. the HMM-based word alignment model of Vogel et al (1996)) can be used to compute this scoring function by aligning each word in a sentence to corresponding word(s) in its pair. A number of scoring functions can be computed using this alignment model.

Khadivi and Ney (2005) suggest training an alignment model on all sentences in the selection pool.[21] The score for each sentence is equal to the average Viterbi alignment probabilities according to source-to-target and target-to-source alignment models, as in Equation (24):

[20] A word-alignment model is a statistical model trained and used to align individual words in a sentence to their translations in the translated sentence. A word-aligned sentence pair contains alignment links used to align words in one sentence to the other (Brown et al (1993)).

[21] In their work, Khadivi and Ney use IBM model 1, HMM and IBM model 2 in succession to train the final model. However, their scoring function does not depend on any particular alignment model, so any alignment model can be used.

$$
\begin{aligned}
f_9\left((s_i)_1^m,(t_i)_1^n\right) &= \frac{1}{m}\log\max_{a_1^m} P_{\mathcal{AM}}\left((s_i)_1^m, a_1^m|(t_i)_1^n\right)\\
&+\frac{1}{n}\log\max_{a_1^n} P_{\mathcal{AM}}\left((t_i)_1^n, a_1^n|(s_i)_1^m\right) \qquad (24)\\
(s_i)_1^m &: \text{words 1 through } m \text{ in sentence } s_i.\\
a_1^m &: \text{target word alignment link for source words 1 through } m.
\end{aligned}
$$

Taghipour et al (2010) introduce **alignment entropy** as a scoring function. Alignment entropy, as defined in Equation (25), is a smooth measure of uniform distribution of alignment links, i.e. given an alignment link count of n, how uncertain the model is about predicting the word corresponding to the link.[22] The highest value for alignment entropy is achieved when all words have the same number of alignment links associated with them.

$$
\begin{aligned}
f_{10}(s_i,t_i) &= \frac{-1}{n}\sum_{(s_i)_1^n} p_1((s_i)_1^n) * \log p_1((s_i)_1^n)\\
&* \frac{-1}{m}\sum_{(t_i)_1^m} p_1((t_i)_1^m) * \log p_1((t_i)_1^m) \qquad (25)\\
p_1((s_i)_k) &= \frac{a((s_i)_k)}{\sum_k a((s_i)_k)}\\
a((s_i)_k) &: \text{The number of links that end on word } (s_i)_k
\end{aligned}
$$

They also use a number of features (Equations (26-29)) inspired by the work of Munteanu and Marcu (2005) on "exploitation of non-parallel corpus for MT" based on the number of alignment links and word fertility:[23]

$$f_{11}(s_i) = N_{src}(s_i) - N_{tgt}(s_i) \qquad (26)$$

$$f_{12}(s_i) = \frac{N_{src}(s_i)}{|s_i|} \qquad (27)$$

$$f_{13}(s_i) = \frac{N_{tgt}(s_i)}{|s_i|} \qquad (28)$$

$$f_{14}(s_i) = \arg\max_{w_i\in s_i} \mathit{fertility}(w_i) \qquad (29)$$

$$
\begin{aligned}
N_{src}(s) &: \text{Number of null aligned source words in sentence } s\\
N_{tgt}(s) &: \text{Number of null aligned target words in sentence } s
\end{aligned}
$$

[22] We think a more natural derivation for alignment entropy of a sentence is averaging alignment entropies of each source or target word according to all possible alignments. This will provide a measure of how uncertain an alignment model is about word alignments in a sentence. This is a more natural use of entropy compared to the uncertainty about a word given the number of Viterbi alignment links for the word.

[23] In word alignment, the fertility of a word is defined as the number of alignment links initiated from that word.

Denkowski et al (2012) add the scoring functions in (30) and (31) to those in the list above:

$$f_{15}(s_i) = \frac{A_{src}(s_i)}{|s_i|} \tag{30}$$

$$f_{16}(s_i) = \frac{A_{tgt}(s_i)}{|s_i|} \tag{31}$$

$A_{src}(s)$: Number of aligned source words in sentence s

$A_{tgt}(s)$: Number of aligned target words in sentence s

Ambati (2011) uses **bidirectional alignment scores** to provide the score for a single alignment in a sentence pair. Denkowski et al (2012) assign a sentence pair score by averaging these alignment scores, as in Equation (32):

$$f_{17}(s_i, t_i) = \sqrt{P_{\mathcal{AM}}(s_i|t_i) * P_{\mathcal{AM}}(t_i|s_i)} \tag{32}$$

In addition, Ambati proposes to use the distance between the bidirectional alignment scores for the highest probability alignment and the second highest probability alignment for a source or target word as a measure of alignment uncertainty. The closer the distance, the more uncertain the alignment model is about the alignment for the word in question (Equation (33)). Although they have not extended this scoring function to the sentence level, an average over all source or target words can be used to provide a sentence-level scoring function.

5.2.3 Other Scoring Functions

We provide a list of sentence-level scoring functions that do not fit the scoring function categories above. These include length-based functions, character-based functions and functions indicating literalness of translation. **Length-based functions** are primarily used to filter out noisy data. In the following length-based functions, sentence length can be calculated as the number of tokens/words or number of characters. A development set can be used to tune thresholds for these length-based functions (Khadivi and Ney (2005); Taghipour et al (2010)). Alternatively, an unsupervised outlier threshold can be developed based on the candidate sentence pool itself (Denkowski et al (2012)).

1. Source sentence to target sentence length ratio and vice versa (Khadivi and Ney (2005); Taghipour et al (2010); Denkowski et al (2012))
2. Difference of source and target sentence lengths (Taghipour et al (2010))
3. Sentence length (Denkowski et al (2012))
4. Length of longest token in the sentence (Denkowski et al (2012))

A few other simple scoring functions can also be used to filter out noisy data.

1. Sentence-end punctuation agreement (Khadivi and Ney (2005))
2. Percentage of alphanumeric characters (Khadivi and Ney (2005); Denkowski et al (2012))
3. Existence of control characters and invalid unicode characters (Denkowski et al (2012))
4. Co-occurrence of special tokens such as email address, URL, date or number on both sides of a sentence pair (Taghipour et al (2010)).

In addition to the functions above, Han et al (2009) introduce two functions to assess lexical and syntactic compatibility of a sentence pair. Their approach is to assign scores to sentence pairs according to the potential for an SMT system to learn from the sentence pair. The idea is that sentence pairs with literal translations are more likely to be useful compared to an abstract or conceptual translation. First, they attempt to estimate the literalness of a translation given a sentence pair using lexical features. This feature is calculated by expanding the words of source and target sentences by a synonym dictionary and then counting the number of words that are translations of each other given a bilingual dictionary. They normalize this value by the total number of words and use it as the function score, as in Equation (33):

$$
\begin{aligned}
f_{18}(s_i,t_i) = {} & \frac{\left|\left(\bigcup_k \underset{src\rightarrow tgt}{\text{BiDic}}\,\underset{src}{\text{Synonyms}}(s_{i_k})\right)\bigcap\left(\bigcup_k \underset{tgt}{\text{Synonyms}}(t_{i_k})\right)\right|}{\left|\bigcup_k \underset{tgt}{\text{Synonyms}}(t_{i_k})\right|} \\
& + \frac{\left|\left(\bigcup_k \underset{src}{\text{Synonyms}}(s_{i_k})\right)\bigcap\left(\bigcup_k \underset{tgt\rightarrow src}{\text{BiDic}}\,\underset{tgt}{\text{Synonyms}}(t_{i_k})\right)\right|}{\left|\bigcup_k \underset{src}{\text{Synonyms}}(s_{i_k})\right|}
\end{aligned}
\tag{33}
$$

$\underset{src\rightarrow tgt}{\text{BiDic}}(w_i)$: Translations for word w_i according to a given dictionary.

$\underset{src}{\text{Synonyms}}(w_i)$: Synonyms for word w_i accoring to a given dictionary.

s_{i_k} : k^{th} word in sentence s_i

In their second approach, Han et al (2009) use part of speech tags to evaluate the grammatical compatibility of a sentence pair. First, they manually map common POS tags of the two languages (e.g. the "noun" POS tag in English is mapped to the "noun" POS tag in Chinese). For a given sentence pair, the number of occurrences of each source tag and its target counterpart tag is calculated. Their distance is computed using the ratio of the smaller count to the larger count. A weighted average[24] of the distance measures for all common tags is used to calculate the cross-lingual grammatical compatibility, as in Equation (34):

[24] The weights are trained using a manually analyzed development set of size 1000.

$$f_{19}(s_i, t_i) = \sum_{tag} \lambda_{tag} \frac{\min\left(\text{Count}(s_i, tag), \text{Count}(t_i, tag)\right) + 1}{\max\left(\text{Count}(s_i, tag), \text{Count}(t_i, tag)\right) + 1} \tag{34}$$

6 Feature Combination and Parameter Tuning

Some data-selection scenarios and scoring functions require parameter tuning.

1. Scoring function parameter(s): Some scoring functions contain a parameter that needs to be tuned (e.g. Han et al (2009)).
2. Feature Combination: Since different scoring functions capture different aspects of the data, it is often desirable to combine more than one function for data selection. For this purpose a combination model (such as linear) is required along with an algorithm to find optimum model parameters (e.g. Khadivi and Ney (2005)).
3. Threshold tuning: data-selection methods that use a threshold to filter/select data, require the threshold parameter to be tuned (e.g. Denkowski et al (2012)).

A common requirement for parameter tuning and feature combination is the availability of a *development set.* In a supervised learning scenario, human annotators are used to create the development set. For example, Han et al use human annotators to label sentence pairs as literal translations versus conceptual translations to create a development set. Since this is an expensive and time-consuming task, alternative techniques for development data creation in semi-supervised learning methods have been developed (Khadivi and Ney (2005); Ananthakrishnan et al (2010b); Taghipour et al (2010)).

1. **Mixing clean and noisy data**: In their work, Khadivi and Ney add noisy data to clean data (labeled accordingly) as the development set to train their noise classification algorithm.
2. **Single Pass Active Learning**: In their effort to train weights for a pairwise sentence comparator classifier, Ananthakrishnan et al (2010a) use a parallel corpus ordered by TER (between the reference translations and the translations produced by the decoder using the models trained on the seed corpus) as their development set. Haffari (2009) uses a similar method to generate a development set to train the feature-combination model weights.
3. **Using Selection Pool**: Denkowski et al (2012) use the candidate selection pool of sentences to compute outlier-detection parameters. They calculate average and standard deviation on the selection pool for a number of features (e.g. sentence-length ratio and alphanumeric density) to filter outliers.

7 Selection Algorithms

Given a constrained optimization problem formulation, consisting of scoring functions, a combination model and size constraints, a selection algorithm is used to solve the optimization problem.

7.1 Threshold-Based Filtering

A context-independent function can be computed for all sentence pairs in one pass and the subset of training data that passes a certain threshold can be selected in linear time. Methods that depend on a threshold tune the threshold prior to filtering. For example, Denkowski et al set the thresholds for length-based filtering at $\operatorname{avg}(s_1^N) \pm 2 * \operatorname{std\,dev}(s_1^N)$. Other selection methods with context-independent scoring functions iteratively select the highest-scoring functions until the constraint is met. These methods require the sentences to be sorted prior to filtering. For a linear filtering time, radix sort can be used. The work of Moore and Lewis is an example of constraint-based filtering as sentences are sorted based on their cross-entropy difference before the filtering is applied. Classification-based approaches such as Khadivi and Ney (2005) and Taghipour et al (2010) follow the same paradigm. Since these approaches combine multiple features, they train a classifier on a development set and then apply the classifier to the selection pool to label sentences as selected versus not selected, or clean versus noisy.

7.2 Greedy Search

The selection method for most methods with context-dependent scoring functions is greedy search. The highest-scoring sentence is selected in each iteration and the models used for the scoring functions are updated. This process is repeated until a specified constraint is met, e.g. the total number of words in selected pool or a minimum quality bar. Eck et al (2005), Haffari et al (2009) and Ananthakrishnan et al (2010a) are examples of this selection method. Due to their high computational complexity, for practical purposes greedy selection algorithms are often paired with batch-learning techniques (see Section 9).

7.3 Submodular Optimization

Kirchhoff and Bilmes (2014) have recently introduced submodularity for data selection in SMT. Submodular functions are a class of set functions that formalize the concept of “diminishing returns”. This makes them a natural theoretical framework for data selection. Formally, a function, $\mathcal{F}$, is called submodular if adding an instance x to a set S increases $\mathcal{F}(S)$ more than (or equal) if it were added to a superset of S, $S \subset S'$, as in Equation (35):

$$\mathcal{F}(S \cup \{x\}) - \mathcal{F}(S) \geq \mathcal{F}(S' \cup \{x\}) - \mathcal{F}(S') \tag{35}$$

In general, solving Equation (3) exactly is NP-complete. However, if the set function $\mathcal{F}$ is submodular the problem can be solved using a greedy algorithm with worst-case solution of $\mathcal{F}(X^*) > (1 - 1/e) * f(X_{opt})$. That is, in the worst case, the function value for the solution from the greedy algorithm is approximately 0.63 of the optimum solution (Kirchhoff and Bilmes (2014)). Moreover, depending on the curvature of the submodular function, the worst-case guarantee can be improved. Graph-based submodular functions are a natural fit for data selection as each sentence is represented as a node in the graph and edge weights are computed based on the similarity of the two sentences. The order of graph construction complexity is quadratic in the number of sentences. This makes this class of submodular functions impractical for large data sets. Another class of submodular functions that are more practical for large data sets is based on bipartite graphs $G = (V, U, E, w)$. In this setting sentences are the left vertices, V. Features (e.g. n-grams) are the right vertices, U, with weight w. Connecting each sentence to its features adds up to the set of edges, E. In this setting, Kirchhoff and Bilmes define a feature-based submodular function as in Equation (36):

$$f(X) = \sum_{u \in U} w_u \phi_u(m_u(X)) \tag{36}$$

Assuming w_u to be positive, $m_u(X)$ to be a non-negative modular function[25] and ϕ_u to be a non-negative, non-decreasing concave function, $\mathcal{F}(X)$ as defined in Equation (36) is submodular. This is a flexible framework for data selection that is scalable to large data sizes. Using this framework requires defining the components below:

1. U: This is the set of features present in all sentences in the selection pool. N-grams are a natural choice for this.
2. $m_u(x)$: This function calculates the relevance of each feature u to sentence x. Kirchhoff and Bilmes use TF-IDF for this function. The main constraint with this function is that it has to be modular.
3. $w(u)$: Each feature can have a weight function to present its importance. In a domain-adaptation task, the frequency of the n-gram in an in-domain development set can be used for this weight function.
4. $\phi_u(a)$: This decay function determines the rate of diminishing returns for redundant instances of a feature u. As this concave function becomes flat, the feature loses its ability to provide additional improvement to the value of a candidate subset. Kirchhoff and Bilmes experiment with square root and logarithmic functions.

[25] A set function is modular if and only if the value of the function over a set equals the sum of the function value over its individual elements.

The greedy algorithm proposed by Kirchhoff and Bilmes is similar to the greedy algorithms mentioned in Section 7.2. Their algorithm chooses the sentence in the sentence pool that, if added to the selected sentences, improves the submodular function's value over the selected sentences the most. This step is repeated until the budget constraint is met.

8 Active Learning

Active learning is a technique used in machine learning where the algorithm can choose the new data from which it learns (Settles (2010)). This technique is often employed in problems where unlabeled data is abundant while the process of obtaining labeled data is expensive or time-consuming. In an active learning framework, the learning algorithm *queries* an *oracle*[26] for *new data points* to be labeled. First, the learning algorithm can obtain *new data points* in several ways.

1. **Pool-based**: The learning algorithm selects data points from a pool of unlabeled data.
2. **Stream-based**: All available unlabeled data is streamed through the learning algorithm and the learning algorithm can choose to query for the data point or discard.
3. **Query Synthesis**: The learning algorithm generates new data points to be labeled by the oracle.

The learning algorithm can use several strategies for selecting new data points such as Uncertainty Sampling, Query by Committee, Expected Model Change, Expected Error Reduction, Variance Reduction and Density-Weighted Methods (Settles (2010)). Active learning selects new data points in an iterative process. In each step, it selects new data point(s), queries the oracle for the label and learns from the results. In contrast, in passive learning, the learning algorithm has no contribution to the data selection and labeling process. From another perspective, in passive learning the data-selection process does not use the learning algorithm to select new data points for labeling. Although the data-selection task for passive learning can also be iterative, the key distinction is that the learning algorithm does not play a role in the data-selection task.

In SMT a data-selection method can be classified as an active learning method only if it follows the iterative data-selection process described above. Context-dependent functions listed in Section 5.1 fit this paradigm. In addition, the selection process must use an element of SMT learning to be considered active learning. Although some prior work specifically refer to their method as active learning Haffari et al (2009); Ambati (2011); Ananthakrishnan et al (2010b), all scoring functions listed in Sections 5.1.2, 5.1.3, 5.1.4 fit the active learning framework.

[26] An oracle can be a human or a human-labeled data set that provides the true label for a query. In the context of MT, the oracle is a human translator or a parallel corpus where the source text has already been translated by humans.

9 Batch-Mode Learning

Batch-mode learning is applicable to both active learning as well as data selection for passive learning. The idea is to select new data points in batches rather than one at a time. Without batch-mode learning, after a single data point is selected and labeled by the oracle, the learning model (e.g. language model or translation model) or the data-selection scoring-function statistic (e.g. n-gram frequency in selected sentences) is updated with the new data point, and all data point scores are recomputed. This process can be very time-consuming, especially if SMT models need to be retrained in order to recompute data point scores. Batch-mode learning addresses this concern by selecting multiple new data points at a time and therefore reducing the number of times the learning model has to be updated and sentence scores need to be recalculated. Using the top-N sentences with highest scores often does not produce the best results since it fails to consider overlap between data points. This is commonly addressed by minimizing overlap between data points within a batch. To select a batch of size K, Ananthakrishnan et al (2010a) go through the data K times.[27] Each time they select the sentence with maximum score and update the scoring function to exclude features (in this case n-grams) used in scoring the selected sentences.

Inspired by the work of Dasgupta and Hsu (2008) on Hierarchical Sampling,[28] Haffari et al (2009) propose Hierarchical Adaptive Sampling for feature combination and batch-learning at the same time. Their work differs from Hierarchical Sampling by not having a predefined static hierarchical cluster over the selection pool. Instead, the hierarchy is created dynamically in steps. Unlike Hierarchical Sampling, their algorithm always selects samples from the cluster node that has been selected for further partitioning. At each step, K sentences are randomly chosen from the selected cluster node and queried for human translations. The selected sentences are added to the set of selected sentence pairs and the SMT models are retrained on the larger set of selected sentence pairs. The two points below are essential to understand the rest of this algorithm.

[27] In this batch-mode learning strategy the idea is to avoid retraining all SMT models after selection of each sentence. A less expensive update is used to improve the diversity of the batch where only the scoring function is updated. After the entire batch is selected, then all SMT models are updated.

[28] Hierarchical Sampling attempts to leverage the cluster structure of the data for sampling in an active learning setting (Dasgupta and Hsu (2008)). In this work, a static hierarchical cluster structure of the unlabeled data is given. A set of cluster nodes for sampling is maintained throughout the algorithm. Initially, the sampling set only contains the root node of the cluster which contains all nodes. Random samples are drawn from the sampling set and queried from the oracle. Based on these queries, each node in the hierarchical cluster maintains statistics about its positive and negative labels. Cluster nodes in the sampling set with mixed labels and more pure child nodes are removed from the sampling set and their child nodes are added. These steps are repeated until all nodes reach a predefined level of purity. This method is motivated by addressing the "sampling bias" problem (Schütze et al (2006)) in active learning and provides theoretical guarantees for a better learning performance than random sampling.

1. Choice of cluster for further partitioning: A cluster node with lowest **average model score** when translated using the updated SMT models is chosen for further partitioning.
2. Partitioning mechanism for a chosen cluster node: The sentences in the partition are sorted according to their **similarity to the selected sentences**. The first α percentage of the sentences are put in the first child cluster node, and the rest in the second child cluster node.

In effect, Hierarchical Adaptive Sampling is combining the "average model score" feature with the "similarity to the selected sentences" feature. While the "average model score" feature selects the cluster that is most difficult to translate, the "similarity to the selected sentences" feature ensures diversity. However, in practice, these two features can be replaced by many of the scoring functions introduced in Section 5. In addition to its high computational complexity, another shortcoming of this approach is that it does not ensure diversity within a single batch. In a follow-up to their work referenced above, Ananthakrishnan et al (2010b) introduce three levels of scoring functions:

1. **Never Updated**: These functions are only computed once for sentences in the selection pool and are never updated. They use n-gram overlap with an in-domain development set as a measure of domain relevance for this purpose.
2. **Updated Per Batch**: These functions are updated after a batch of sentences are selected and queried for human translation. They compute translation difficulty by retraining the SMT models with the new batch and derive translation difficulty features based on the performance of new models for the next batch of sentences.
3. **Updated Per Sentence**: These functions are recomputed every time a single sentence is added to the pool. N-gram overlap with existing sentences in the batch (batch diversity) is used for this purpose and has to be updated per sentence.

Although only batch diversity is recomputed after the selection of each sentence, Ananthakrishnan et al use all three functions above to select the highest-scoring sentence in each step. This is continued until the predefined batch size is reached.

10 Evaluation Methods

All prior work evaluate the effectiveness of data-selection methods by training on a selected parallel corpus and testing on a randomly held-out test set. In data selection for domain-adaptation scenarios (Moore and Lewis (2010); Axelrod et al (2011)) the test set is an in-domain test set. As for most SMT tasks, BLEU is the most common evaluation method. While in some cases selecting a subset of the data outperforms cases using the entire data set (Eck et al (2005); Kirchhoff and Bilmes (2014)), the general trend is that selecting more data improves evaluation results. For this reason, comparison between different selection methods is often done by running each selection algorithm multiple times. Each time the selection constraint (e.g. maximum number of words selected) is set to a different percentage of the data. An SMT system is trained on each selected subset and tested on the held-out test set. This provides several comparison points between the selection methods. The objective in cost-focused selection methods is to meet a predefined quality bar at minimum cost (Bloodgood and Callison-Burch (2010)). In these scenarios, different selection methods are compared based on their cost to reach a predefined BLEU score.

11 Comparative Summary of Cited Work

The work of Eck et al (2005) is noteworthy as the first work published on data selection for SMT. Features introduced in this work such as *n*-gram coverage and TF-IDF are still the basis for much of the most recent work. A practical extension of their work is the Vocabulary Saturation Filter (Lewis and Eetemadi (2013)) which relaxes the dependency on previously selected sentences; this allows for linear-time application of *n*-gram coverage to very large data sets. Another extension to the work of Eck et al is using feature-decay functions (Ambati et al (2010); Biçici et al (2014, 2015)) to implement the idea of diminishing returns. While in their original work, Eck et al maintained *n*-gram count statistics, other work (Haffari et al (2009); Liu et al (2009)) used language models for more accurate probability estimation instead.

Moore and Lewis (2010) introduce cross-entropy difference which has become one of the most commonly used approaches in data selection in the domain adaptation setting. Bilingual cross-entropy difference is a natural extension of this work that takes both sides of the parallel data into consideration (Axelrod et al (2011)).

The work of Haffari et al (2009) is uniquely objective in that they use the performance of a translation system trained on the selected data in translating new data as a measure of translation difficulty and thus the usefulness of new data. Their approach to active learning is computationally intensive. In their first extension to this work, Ananthakrishnan et al (2010a) address the computational intensity problem by training a classifier to compare sentences based on their translation difficulty. In a follow up work they combine the approaches of Eck et al, Moore and Lewis and Haffari et al to develop a multi-layer active learning strategy that combines translation difficulty, domain appropriateness and diversity into a single framework (Ananthakrishnan et al (2010b)).

The work of Kirchhoff and Bilmes (2014) offers the most effective data-selection search framework with theoretical guarantees. Although they do not introduce any new features in their submodular optimization method for data selection, they offer a flexible framework where custom functions can be used for sentence features, relevance scores, feature weights and a decay function.

The work of Ambati et al (2010) is unique in that they introduce a customized active learning strategy for crowdsourcing while introducing new features such as phrasal entropy and KL-divergence. Bloodgood and Callison-Burch (2010) offer a cost-focused active-learning strategy by asking a crowd for translation of specific phrases within a sentence instead of full sentence translation. Using this method they are able to outperform all previous approaches on a per-word translation pricing scheme.

Khadivi and Ney (2005) and Taghipour et al (2010) use a classifier-based approach to filter out noisy data while Denkowski et al (2012) use simple average and standard deviation statistics to achieve the same goal.

Han et al (2009) have a unique approach to data selection where they use lexical and syntactic compatibility between source and target sentences to estimate the literalness of the translation. Their idea is to select more literal translations as an SMT system can only learn from literal translations and would not benefit from free translations.

Finally the work of Kauchak (2006) is unique as they provide the most objective measure of sentence pair usefulness for SMT, although their method is not practical for any real-world scenario. In their work, they create 500 random subsets from the selection pool to train 500 different sets of models. The usefulness of each sentence pair is estimated based on its membership in training data for different model sets and the model sets' performance.

Related work in the literature is included in Table 2 according to the main components of training data selection listed.

12 Related Research Areas

Data selection in MT is closely related to **quality estimation**. In a quality estimation task, given a sentence and its translation using an SMT decoder, the quality of the translation is estimated without a reference translation (Specia et al (2010)). This is a very similar task to data selection. In fact many features used in data selection have been inspired by features used in quality estimation (Denkowski et al (2012)).

Data selection is a practical method of adapting SMT models to a domain and is often used in **domain adaptation** tasks. Other domain adaptation techniques include translation model adaptation (Chen et al (2014)).

Most of the **active learning** literature in machine learning is focused on classification tasks with a limited number of features. Therefore, the focus is on selecting data points that are most useful in learning feature weights. Active learning for MT is unique in that the number of features to be learned is very large or, for all practical purposes, unlimited for large data sets. The active learning task in this case is to identify data points with the most useful features in addition to finding data points that are the most useful in estimating the weights for the features. This makes active learning for MT unique in the active learning literature.

Co-training and **self-training** are also closely related areas to data selection for MT (Yarowsky (1995); Haffari (2009)). Both techniques are used when labeled data is scarce and unlabeled data is abundant. In a simple co-training

Reference	Scenario	Features	Algorithm
Eck et al (2005)	LR	*n*-gram Coverage TF-IDF	Greedy Search
Haffari et al (2009)	LR	Round-Trip Translation Accuracy Phrase Pair Utility *N*-gram Utility *N*-gram Coverage Decoder Translation Model Score	Greedy Search Hierarchical–Adaptive Sampling
Mandal et al (2008)	LR	Inter-System Disagreement Language Models' Perplexity Ratio	Greedy Search Threshold Filtering
Lü et al (2007)	DA	TF-IDF	Threshold Filtering
Taghipour et al (2010)	NR	Word Alignment Probability Word to Null Alignment Count Word to Null Alignment Ratio Alignment Entropy Top-3 Word Fertilities Factoid Co-Appearance Sentence Length Ratio LM Probability Difference LM Probability Ratio	Classifier
Denkowski et al (2012)	NR	Language Model Score Combined Alignment Score Length Ratio Alphanumeric Intensity Maximum Token Length	Threshold Filtering
Ananthakrishnan et al (2010a)	LR	Expected Translation Error–Reduction	Greedy Search Classification
Han et al (2009)	NR	Lexical Compatibility Grammatical Compatibility	Threshold Filtering
Wei et al (2013)	TR	TF-IDF	Submodular-Optimization
Liu et al (2010a)	TR	Weighted Phrase Entropy Weighted Sub-tree Entropy	Greedy Search
Moore and Lewis (2010)	DA	Cross Entropy Difference	Threshold Filtering
Chao and Li (2011b)	TR	*N*-gram Co-Occurrence	Greedy Search
Axelrod et al (2012)	DA	Cross Entropy Difference	Threshold Filtering
Gangadharaiah et al (2009)	TR	Average Alignment Probability	Greedy Search
Ambati (2011)	LR	Phrase Probability Unseen *n*-grams	Greedy Search
Khadivi and Ney (2005)	NR	Sentence Length Ratio Alphanumeric Character Presence Sentence Ending Symbol Similarity Automatic Language Identification Alignment Probability	Classifier
Bloodgood and Callison-Burch (2010)	QI	Unseen *N*-Grams	Threshold Filtering
Kauchak (2006)	LR	Estimated Contribution Score	Threshold Filtering Greedy Search
Okita (2009)	NR	Minimum *N*-Gram Occurrence Sentence Length	Greedy Search
Allauzen et al (2011)	NR	LM Perplexity	Threshold Filtering
Yasuda et al (2008)	DA	LM Perplexity	Threshold Filtering
Eetemadi and Radha (2010)	QI	*N*-gram Co-Occurrence	Greedy Search
Lewis and Eetemadi (2013)	TR	Minimum *N*-Gram Occurrence Alignment Probability	Threshold Filtering

Table 2: Related Work. LR: Low-Resource Language; DA: Domain Adaptation; NR: Noise Reduction; TR: Training Resource Reduction; QI: Quality Improvement.

setting, two classifiers trained on labeled data classify the same unlabeled data set using two different but complimentary feature sets. If the classifiers agree in their labels, the data points and their labels are added as training data and classifiers are retrained. In self-training, unlabeled data is labeled using a single classifier trained on the limited labeled data. Data points with high classification confidence are added to the training as new data points. These techniques have been leveraged in data selection for MT as well. In his work, Haffari experiments with self-training while the work of Mandal et al (2008) uses features that are closely related to co-training.

Finally, another related research area is **model pruning** in SMT. In data-selection scenarios with model size deployment constraints, model pruning can be used as an alternative to data selection. Since model pruning is performed after model training, it is only useful with deployment resource constraints (not training resource constraints). The research in this area focuses on **phrase table pruning** (Wuebker et al (2010)) and **language model pruning** (Goodman and Gao (2000)).

13 Summary

In practice, performing a data-selection task for SMT requires addressing two technical questions:

1. **Scoring Function:** How is the value of a sentence or sentence pair objectively evaluated?
2. **Selection Algorithm:** Given a scoring function and a constraint, how is the optimum data subset selected?

In this paper we reviewed a long list of objective functions along with a number of selection algorithms. This combination offers an overwhelming number of different configurations to perform a data-selection task. However, in practice, the number of applicable solutions can be narrowed down significantly by the application scenario (Section 2) and data size.

Multiple data-selection scenarios are often employed in the development of a single SMT system. For example, when developing a general-purpose MT service for English↔French, there are three different issues with training data that can be addressed using data selection.

1. Much of the available parallel data is collected through web crawling and is noisy.
2. There is too much parallel data to train and iterate on in a timely manner (Lewis and Eetemadi (2013)).
3. Even if we could train on all the data, if we were to deploy the SMT system on an offline mobile device, the model sizes would be too large.

In this scenario the data can be first cleaned using context-independent scoring functions listed in Equations (19)-(32) and Section 5.2.3. Investing in creating a development set to train a classifier (Taghipour et al (2010)) would enable training weights for multiple scoring functions. If few scoring functions are used, using average and standard deviation for threshold-based filtering (Denkowski et al (2012)) can be effective as well.

The noise level of a sentence pair can be estimated independently using context-independent scoring functions. However, when there is too much clean data, selecting a subset of the data that can perform the best on a blind test set is more complicated. Active learning provides an ideal solution as it is the most objective. On the other hand it requires retraining a full SMT system on every iteration which makes it impractical for large data sets. Active learning inherently uses context-dependent scoring functions since it uses models built from the selected pool to evaluate new sentences from the selection pool. Other context-dependent scoring functions (Eck et al (2005)) use incremental statistics from the selected pool to score new sentences in the selection pool. A general and efficient framework for the use of context-dependent scoring functions is submodular optimization (Kirchhoff and Bilmes (2014)). For very large data sets that a computational complexity of $O(N \times \log N)$ is not practical, using a Vocabulary Saturation Filter (Lewis and Eetemadi (2013)) offers a linear solution. The deployment resource restriction scenario is similar to the training resource-limitation scenario with the exception that phrase table or language model pruning is a viable alternative to data selection.

Another common scenario for the data-selection task in SMT is developing an SMT system for low-resource languages. In this scenario there is a small parallel seed corpus and a large monolingual selection pool. The idea is to select a limited number of sentences from the selection pool to be manually translated by humans and added to the seed corpus. In this scenario active learning strategies have proved to be the most successful (Haffari et al (2009); Ananthakrishnan et al (2010b); Ambati (2011)). The active learning strategies can be used with any of the context-dependent scoring functions reviewed in Section 5.1, while batch-mode learning strategies reviewed in Section 9 can be used to speed up the data-selection process.

Acknowledgements We thank reviewers and in particular the editor who went above and beyond in helping us make this a better paper.

References

Adesam Y (2012) The multilingual forest : Investigating high-quality parallel corpus development. PhD thesis, Stockholm University, Stockholm, Sweden

Allauzen A, Bonneau-Maynard H, Le HS, Max A, Wisniewski G, Yvon F, Adda G, Crego JM, Lardilleux A, Lavergne T, Sokolov A (2011) LIMSI @ WMT11. In: Proceedings of the Sixth Workshop on Statistical Machine Translation, Edinburgh, UK, pp 309–315

Ambati V (2011) Active learning and crowdsourcing for machine translation in low resource scenarios. PhD thesis, Carnegie Mellon University, Pittsburgh, PA

Ambati V, Vogel S, Carbonell J (2010) Active learning and crowd-sourcing for machine translation. In: Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), Valletta, Malta, vol 7, pp 2169–2174

Ananthakrishnan S, Prasad R, Stallard D, Natarajan P (2010a) Discriminative sample selection for statistical machine translation. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Cambridge, MA, pp 626–635

Ananthakrishnan S, Prasad R, Stallard D, Natarajan P (2010b) A semi-supervised batch-mode active learning strategy for improved statistical machine translation. In: Proceedings of the Fourteenth Conference on Computational Natural Language Learning, Uppsala, Sweden, pp 126–134

Axelrod A, He X, Gao J (2011) Domain adaptation via pseudo in-domain data selection. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK, pp 355–362

Axelrod A, Li Q, Lewis W (2012) Applications of data selection via cross-entropy difference for real-world statistical machine translation. In: Proceedings of the International Workshop on Spoken Language Translation, Hong Kong, pp 201–108

Banerjee P, Naskar S, Roturier J, Way A, van Genabith J (2011) Domain adaptation in statistical machine translation of user-forum data using component level mixture modelling. In: Proceedings of Machine Translation Summit XIII, Xiamen, China, pp 285–292

Biçici E, Yuret D (2011) Instance selection for machine translation using feature decay algorithms. In: Proceedings of the Sixth Workshop on Statistical Machine Translation, Association for Computational Linguistics, Edinburgh, UK, pp 272–283

Biçici E, Liu Q, Way A (2014) Parallel FDA5 for fast deployment of accurate statistical machine translation systems. In: Proceedings of the Ninth Workshop on Statistical Machine Translation, Baltimore, MD, pp 59–65

Biçici E, Liu Q, Way A (2015) Parfda for fast deployment of accurate statistical machine translation systems, benchmarks, and statistics. In: Proceedings of the Tenth Workshop on Statistical Machine Translation, Lisbon, Portugal, pp 74–78

Bloodgood M, Callison-Burch C (2010) Bucking the trend: Large-scale cost-focused active learning for statistical machine translation. In: ACL 2010, The 48th Annual Meeting of the Association for Computational Linguistics, Conference Proceedings, Uppsala, Sweden, pp 854–864

Brown PF, Della Pietra VJ, Della Pietra SA, Mercer RL (1993) The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics 19(2):263–311

Callison-Burch C, Dredze M (2010) Creating speech and language data with amazon's mechanical turk. In: Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Montreal, Canada, pp 1–12

Chao W, Li Z (2011a) A graph-based bilingual corpus selection approach for SMT. In: Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation, Singapore, pp 120–129

Chao W, Li Z (2011b) Improved graph-based bilingual corpus selection with sentence pair ranking for statistical machine translation. In: Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence, Boca Raton, FL, USA, pp 446–451

Chen B, Kuhn R, Foster G (2014) A comparison of mixture and vector space techniques for translation model adaptation. In: AMTA 2014, Proceedings of the 11th Conference of the Association for Machine Translation in the Americas, Vol. 1: MT Researchers Track, Vancouver, BC, pp 124–138

Chiang D (2005) A hierarchical phrase-based model for statistical machine translation. In: 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, MI, pp 263–270

Cover TM, Thomas JA (1991) Elements of Information Theory. Wiley-Interscience, New York, NY, USA

Dasgupta S, Hsu D (2008) Hierarchical sampling for active learning. In: Proceedings of the 25th international conference on Machine learning, Helsinki, Finland, pp 208–215

Denkowski M, Hanneman G, Lavie A (2012) The CMU-Avenue french-english translation system. In: Proceedings of the NAACL 2012 Workshop on Statistical Machine Translation, Montreal, Canada, pp 261–266

Dyer C, Cordova A, Mont A, Lin J (2008) Fast, easy, and cheap: Construction of statistical machine translation models with mapreduce. In: Proceedings of the Third Workshop on Statistical Machine Translation, Columbus, OH, pp 199–207

Eck M, Vogel S, Waibel A (2005) Low cost portability for statistical machine translation based in N-gram frequency and TF-IDF. In: IWSLT 2005, Proceedings of the International Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation, Pittsburgh, PA, USA, 7pp.

Eetemadi S, Radha H (2010) Effects of parallel corpus selection on statistical machine translation quality. In: NW-NLP 201: Proceedings of the Pacific Northwest Regional NLP Workshop, Redmond, WA, 2pp.

Gangadharaiah R, Brown R, Carbonell J (2009) Active learning in example-based machine translation. In: Proceedings of the 17th Nordic Conference of Computational Linguistics NODALIDA, Odense, Denmark, pp 227–230

Goodman J, Gao J (2000) Language model size reduction by pruning and clustering. In: Proceedings of the International Conference on Spoken Language Processing, Beijing, China, pp 110–113

Goutte C, Carpuat M, Foster G (2012) The impact of sentence alignment errors on phrase-based machine translation performance. In: Proceedings of the 10th Conference of Association for Machine Translation in the Americas, San Diego, CA, 8pp.

Haffari G (2009) Machine learning approaches for dealing with limited bilingual training data in statistical machine translation. PhD thesis, Simon

Fraser University, Burnaby, BC, Canada

Haffari G, Roy M, Sarkar A (2009) Active learning for statistical phrase-based machine translation. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, CO, pp 415–423

Han X, Li H, Zhao T (2009) Train the machine with what it can learn: corpus selection for SMT. In: BUCC-09, Proceedings of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora, Singapore, pp 27–33

Jiang J, Way A, Carson-Berndsen J (2010) Lattice score-based data cleaning for phrase-based statistical machine translation. In: Proceedings of the 14th Annual Conference of the European Association for Machine Translation, Saint-Raphaël, France, 8pp.

Kauchak D (2006) Contributions to research on machine translation. PhD thesis, UC San Diego, San Diego, CA

Khadivi S, Ney H (2005) Automatic filtering of bilingual corpora for statistical machine translation. Natural Language Processing and Information Systems 3513:263–274

Kirchhoff K, Bilmes J (2014) Submodularity for data selection in machine translation. In: EMNLP 2014, The 2014 Conference on Empirical Methods In Natural Language Processing, Proceedings of the Conference, Doha, Qatar, pp 131–141

Koehn P (2005) Europarl: a parallel corpus for statistical machine translation. In: Proceedings of MT Summit X, Phuket, Thailand, pp 79–86

Koehn P (2009) Statistical machine translation. Cambridge University Press, Cambridge, UK

Koehn P, Och F, Marcu D (2003) Statistical Phrase-Based Translation. In: Proceedings of the Joint Human Language Technology Conference and the Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL), Edmonton, Canada, pp 127–133

Kullback S, Leibler L (1951) On information and sufficiency. Annals of Mathematical Statistics 22(1):79–86

Lewis W, Eetemadi S (2013) Dramatically reducing training data size through vocabulary saturation. In: Proceedings of the Eighth Workshop on Statistical Machine Translation, Sofia, Bulgaria, pp 281–291

Liu P, Zhou Y, Zong C (2009) Approach to selecting best development set for phrase-based statistical machine translation. In: Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation, Hong Kong, pp 325–334

Liu P, Zhou Y, Zong C (2010a) Data selection for statistical machine translation. In: Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE), Beijing, China, pp 1–5

Liu P, Zhou Y, Zong CQ (2010b) Approaches to improving corpus quality for statistical machine translation. In: Proceedings of the International Conference on Machine Learning and Cybernetics, Qingdao, China, vol 6, pp 3293–3298

Lü Y, Huang J, Liu Q (2007) Improving statistical machine translation performance by training data selection and optimization. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, pp 343–350

Mandal A, Vergyri D, Wang W, Zheng J, Stolcke A, Tur G, Hakkani-Tur D, Ayan NF (2008) Efficient data selection for machine translation. In: Proceedings of the Spoken Language Technology Workshop, 2008, Goa, India, pp 261–264

Manning CD, Schütze H (1999) Foundations of statistical natural language processing. MIT press, Cambridge, MA

Moore RC, Lewis W (2010) Intelligent selection of language model training data. In: ACL 2010, The 48th Annual Meeting of the Association for Computational Linguistics, Conference Proceedings, Uppsala, Sweden, pp 220–224

Munteanu DS, Marcu D (2005) Improving machine translation performance by exploiting non-parallel corpora. Computational Linguistics 31(4):477–504

Okita T (2009) Data cleaning for word alignment. In: Proceedings of the ACL-IJCNLP 2009 Student Research Workshop, Singapore, pp 72–80

Okita T, Naskar SK, Way A (2009) Noise reduction experiments in machine translation. In: ECML-PKDD, Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, Bled, Slovenia, 2pp.

Ozdowska S, Way A (2009) Optimal bilingual data for french-english pb-smt. In: Proceedings of EAMT-09, the 13th Annual Conference of the European Association for Machine Translation, Barcelona, Spain, pp 96–103

Papineni K, Roukos S, Ward T, Zhu WJ (2002) Bleu: A method for automatic evaluation of machine translation. In: ACL-2002: 40th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Philadelphia, PA, pp 311–318

Pecina P, Toral A, Papavassiliou V, Prokopidis P, Tamchyna A, Way A, Van Genabith J (2014) Domain adaptation of statistical machine translation using web-crawled resources and model parameter tuning. Language Resources and Evaluation 49(1):147–193

Resnik P (1999) Mining the web for bilingual text. In: ACL-1999: 37th Annual Meeting of the Association for Computational Linguistics: Proceedings of the conference, College Park, Maryland, USA, pp 527–534

Schütze H, Velipasaoglu E, Pedersen JO (2006) Performance thresholding in practical text classification. In: Proceedings of the 15th ACM international conference on Information and knowledge management, Kansas City, MO, pp 662–671

Settles B (2010) Active learning literature survey. Tech. rep., University of Wisconsin, Madison, WN, URL `http://burrsettles.com/pub/settles.activelearning.pdf`

Snover M, Dorr B, Schwartz R, Micciulla L, Makhoul J (2006) A study of translation edit rate with targeted human annotation. In: AMTA 2006: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas, Visions for the Future of Machine Translation, Cambridge, Massachusetts, USA, pp 223–231

Somers H (2005) Round trip translation: What is it good for? In: ALTW 2005: Proceedings of Australasian Language Technology Workshop, Sydney, Australia, pp 127–133

Specia L, Raj D, Turchi M (2010) Machine translation evaluation versus quality estimation. Machine Translation 24(1):39–50

Taghipour K, Afhami N, Khadivi S, Shiry S (2010) A discriminative approach to filter out noisy sentence pairs from bilingual corpora. In: Proceedings of the 5th International Symposium on Telecommunications, Tehran, Iran, pp 537–541

Ueffing N, Ney H (2007) Word-level confidence estimation for machine translation. Computational Linguistics 33(1):9–40

Vogel S, Ney H, Tillmann C (1996) HMM-based word alignment in statistical translation. In: Proceedings of the 16th Conference on Computational Linguistics - Volume 2, Copenhagen, Denmark, pp 836–841

Wei K, Liu Y, Kirchhoff K, Bilmes J (2013) Using document summarization techniques for speech data subset selection. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, GA, pp 721–726

Wuebker J, Mauser A, Ney H (2010) Training phrase translation models with leaving-one-out. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, pp 475–484

Yarowsky D (1995) Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics, Cambridge, Massachusetts, USA, pp 189–196

Yasuda K, Zhang R, Yamamoto H, Sumita E (2008) Method of selecting training data to build a compact and efficient translation model. In: Proceedings of the Third International Joint Conference on Natural Language Processing, Hyderabad, India, vol 2, pp 655–660

Zens R, Stanton D, Xu P (2012) A systematic comparison of phrase table pruning techniques. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju, Korea, pp 972–983