# Data Selection for IT Texts using Paragraph Vector

**Mirela-Stefania Duma and Wolfgang Menzel**
University of Hamburg
Natural Language Systems Division
{mduma, menzel}@informatik.uni-hamburg.de

## Abstract

This paper presents an overview of the system submitted by the University of Hamburg to the IT domain shared translation task as part of the ACL 2016 First Conference of Machine Translation (WMT 2016). We have chosen data selection as a domain adaptation method. The filtering of the general domain data makes use of paragraph vectors as a novel approach for scoring the sentences. Experiments were conducted for English-German under the constrained condition.

## 1 Introduction

The WMT 2016 shared task of translating IT documents focuses on translation of answers in a cross-lingual help-desk service. This paper describes the system submitted by the University of Hamburg to this task. We took part in the English-German translation track in which twelve systems (seven constrained and five unconstrained ones) from four different organizations participated. The challenges for this task came from the fact that the available in-domain data for the constrained condition is very small. Moreover, the in-domain differs considerably from any of the domains of the given general domain data.

We propose a method of data selection by filtering the general domain data applying a threshold on the similarity between vector representations for the sentences from the general domain and the in-domain. Sentences are described by paragraph vectors which are trained together with word vectors in order to predict the upcoming words within that paragraph (Le and Mikolov, 2014). Given a sentence from the general domain, our procedure identifies a set of candidate sentences that are most similar to the reference. If at least one of the retrieved sentences comes from the in-domain then the general domain sentence is considered similar to the in-domain, otherwise it is discarded. This binary decision has the advantage that only one MT system needs to be trained and the disadvantage that it gives only a fixed ratio of general domain data to be kept depending on the chosen threshold.

In order to overcome the disadvantage that the paragraph vector method has, we extend it from using a binary decision filtering to scoring and ranking all the sentences from the general domain from which a certain amount of training sentences can be selected. This extended version is a prerequisite for being able to train and compare multiple MT systems using different ratios of data to be kept.

We first summarize related work in data selection for Statistical Machine Translation (SMT) in Section 2, then describe Paragraph Vector that we used for our data selection method in Section 3. Section 4 presents the experimental settings of the submitted systems and section 5 contains an overview of their performance in the shared IT task.

## 2 Related work

A range of different methods for domain adaptation of models for statistical machine translation have been developed including mixture modeling, instance weighting, transductive learning, or data selection (Chen et al., 2013).

The data selection approach is the focus of this paper. In the state of the art, data selection is used at the corpus-level, where the selected data is joined together, or at the model-level, where several models are combined together in the translation phase (Wang et al., 2013a). The main workflow of the data selection method consists of the



*Proceedings of the First Conference on Machine Translation*, Volume 2: Shared Task Papers, pages 428–434,
Berlin, Germany, August 11-12, 2016. 

following steps:

- scoring: a measure is used to determine how similar the sentences from the general domain are to the in-domain
- filtering: sentences from the general domain are selected, if their similarity score is greater than a predefined threshold.
- training: the selected sentences are used as additional training data to develop the language model, to weight the phrase pairs or for tuning purposes.

To compute the similarity score three approaches are commonly used: information retrieval inspired, perplexity-based and edit distance similarity inspired.

*TF-IDF*[1] term weighing as used in information retrieval was adopted by (Hildebrand et al., 2005) where each sentence from the source side of the bilingual training data constitutes one document (represented using *TF-IDF*) and each sentence from the test data is used as a query. The cosine distance similarity is used to compute the relevance of the queries to the documents. Lü et al. (2007) also uses the cosine to select sentences for offline and online training data optimization. Tamchyna et al. (2012) presents a method where sentences are extracted from the general domain by translating the source side of a test set and using it in computing the cosine similarity to the general domain.

In Mandal et al. (2008) and in Axelrod et al. (2011) language model perplexity was used to score sentences. Foster et al. (2010) used phrase pairs instead of sentences and learned weights for them using in-domain features based on word frequencies and perplexities. In Mansour et al. (2011), the cross-entropy score is used for language model filtering together with a translation model score that estimates the likelihood that a source and a target sentence are a translation of each other. Toral et al. (2015) introduced linguistic information such as lemmas, named entities and part-of-speech tags into the preprocessing of the data and then ranked the sentences by perplexity.

The edit distance which computes the minimum number of edits needed to transform a sentence from the general domain into a sentence from the in-domain was used in Wang et al. (2013b). A combination of the three data selection approaches is presented in Wang et al. (2013a, 2013c).

We propose a new approach of filtering general domain sentences using paragraph vectors (Le and Mikolov, 2014) to determine sentence similarity in a high-dimensional vector space. To the knowledge of the authors, this is the first time Paragraph vector is applied to data selection for SMT.

## 3 Paragraph vector

In this section we describe Paragraph vector (Le and Mikolov, 2014) which stands at the core of the proposed data selection method. It has been successfully employed in sentiment detection and information retrieval tasks. Le and Mikolov (2014) propose an unsupervised framework that learns continuous distributed vector representations for phrases, sentences or documents.

The idea of learning paragraph vectors is similar to the approach used in learning word vectors (Mikolov et al., 2013): word vectors are used in predicting a word given its sentential context and paragraph vectors adopt the same idea to contexts sampled from a paragraph.

The model maps context words and a paragraph identifier to the word that is going to be predicted. The contexts have a fixed length and are sampled from a sliding window over the paragraph. The mapping is established by means of two matrices: one consisting of the trained paragraph vectors and the other consisting of word vectors. The paragraph vector is shared among all the contexts sampled from the same paragraph (but not among all paragraphs). The word vectors are shared between all the paragraphs. Paragraph and word vectors are combined during training and inference either by concatenation or by averaging. The paragraph and word vectors are trained on pairs consisting of the word to be predicted and a sampled context tagged by a paragraph identifier. (Le and Mikolov, 2014)

We use single sentences as paragraphs. The reason why we adopted Paragraph vector is because they reflect semantic relatedness, similar to word vectors. Moreover, we have chosen paragraph vectors for representing sentences as vectors because the approach does not require tuning, parsing or availability of labeled data. The implementation of paragraph vectors we used is Doc2vec from the *gensim* toolkit[2] (Řehůřek and Sojka, 2010).

[1] Term frequency - Inverse document frequency

[2] `https://radimrehurek.com/gensim/`

## 4 Experiments

For all the submitted systems, we used only the data distributed for the shared IT task. For the general domain training data we chose Commoncrawl[3] (made available by WMT) because it is a relatively large corpus and contains crawled data from a variety of domains including the IT domain. As in-domain training data we concatenated the corpora provided by the task. We tuned the systems with 2000 sentences from Batch1a and Batch2a provided by the shared task and evaluated them on Batch3a.

Our systems have been developed using the Moses phrase-based MT toolkit (Koehn et al., 2007) and the Experiment Management System (Koehn, 2010) that facilitates the preparation of scripts for experiments.

### 4.1 Data preprocessing

All the available data were tokenized, cleaned (i.e. restricted to a maximum sentence length of 80 words) and lowercased. The general domain data was filtered by removing the sentence pairs that do not pertain to the English-German language pair as well as sentences that contain non-alpha characters. In addition to that, punctuation was normalized using the *normalize-punctuation.perl* script. Approximately 25K sentences were removed because they were not considered English-German sentence pairs by the jlangdetect library[4] and further 650 sentences have been discharged because they contained non-alpha characters. Table 1 presents some data statistics for both domains after preprocessing:

| **Corpora** | **Sentences** | **Tokens** | |
|---|---|---|---|
| | | English | German |
| Commoncrawl | 2.34M | 50.33M | 46.11M |
| IT | 210K | 1.48M | 1.44M |

Table 1: Corpora statistics after preprocessing

### 4.2 Experimental settings

We performed word alignment using GIZA++ (Och and Ney, 2003) with the default *grow-diag-final-and* alignment symmetrization method. For the language model (LM) estimation we trained 5-gram LMs using the SRILM toolkit (Stolcke, 2002) with Kneser-Ney discounting (Kneser and Ney, 1995) on the target side of the Commoncrawl and IT corpora. When LM interpolation was needed, the in-domain LM and the general domain LM were interpolated using weights tuned to minimize the perplexity on the tuning set. The same data was used for tuning the systems with MERT (Och, 2003).

For the BLEU-cased scores training recasing was performed using the default configuration from the EMS script: language model trained using KenLM (Heafield, 2011) and order 3. Due to time limitations, we did not try to further improve the recaser model.

### 4.3 Baselines

The baseline system $UHBS\_simple$ was trained on the concatenation of the in-domain data and the complete general domain data. The second baseline, $UHBS\_lmi$, only differed from $UHBS\_simple$ in its language model that was created by LM interpolation. The motivation for training a second, i.e. stronger baseline, is that we intended to compare the translation results of the system submitted to the competition ($UHDS\_doc2vec$) with the one produced by a competitive approach.

### 4.4 Data selection using Doc2vec

In this section the submitted system $UHDS\_doc2vec$ is described. The filtering procedure receives as input the bilingual in-domain corpus $\mathcal{I}n$, the bilingual general domain $\mathcal{G}en$, the number of most similar sentences $\mathcal{N}$ that should be retrieved given a threshold $\delta$ that will be described later. Our approach is monolingual as we used only the source side of the corpus data to select sentences from the general domain corpus. To train the paragraph vectors we concatenated $\mathcal{I}n$ and $\mathcal{G}en$ resulting in the data set $\mathcal{C}$. Training the doc2vec model required tagging every sentence from the source side of the concatenated corpus $\mathcal{C}_{source}$ with its corresponding line number in the corpus and building a vocabulary from the tagged $\mathcal{C}$. Therefore, a sentence that came from $\mathcal{I}n$ was tagged with a number from $[1, size_{In}]$ and a sentence that came from $\mathcal{G}en$ was tagged with a number from $[size_{In} + 1, size_{In} + size_{Gen}]$.

The doc2vec model was trained on the tagged $C_{source}$. After obtaining the doc2vec model $\mathcal{M}$, the algorithm iterates through every sentence pair

models/doc2vec.html

[3] http://commoncrawl.org/

[4] https://github.com/melix/jlangdetect

**Algorithm 1** Doc2vec Filtering

1: **procedure** FILTER($\mathcal{I}n, \mathcal{G}en, \mathcal{N}, \delta$)
2: $\mathcal{C} \leftarrow \mathcal{I}n + \mathcal{G}en$
3: **for each** sentence $s_i \in \mathcal{C}_{source}$ **do**
4: tag $s_i$ with the line number $i$
5: build vocabulary from tagged $\mathcal{C}_{source}$
6: train doc2vec model $\mathcal{M}$ using tagged $\mathcal{C}_{source}$
7: **for each** sentence pair $(s_i, t_i) \in \mathcal{G}en$ **do**
8: $\mathcal{R}_i = top(\mathcal{N}, mostSimilar(\mathcal{M}, s_i))$
9: $Sim_{s_i} = \{(index, score) \in \mathcal{R}_i |\ index \in [1, size_C], score \in (0, 1)\}$
10: **if** $\exists (index, score) \in Sim_{s_i} : (index < size_{In}, score > \delta)$ **then**
11: add $(s_i, t_i)$ to *FilteredCorpus*

Figure 1: Doc2vec filtering algorithm

from $\mathcal{G}en$. Given a sentence pair $(s_i, t_i) \in \mathcal{G}en$, the top $\mathcal{N}$ most similar vectors to $s_i$ are retrieved in the form of a pair $(index, score)$ where $index$ gives the tag (i.e. the line number) of the selected similar sentence to $s_i$ and $score$ specifies the similarity between $s_i$ and $s_{index}$. The similarity is computed as the cosine between the two vectors.

The list of top $\mathcal{N}$ most similar sentences for each sentence from $Gen$ is now filtered by comparing them to a prespecified threshold $\delta$ creating a reduced data set $FilteredCorpus$. A sentence pair $(s_i, t_i)$ is included into $FilteredCorpus$ if at least one pair $(index, score)$ originates from the in-domain ($index < size_{In}$) and has a $score > \delta$. With a value setting of $\delta = 0.5$ we selected 47% of the sentences of $Gen$. Systematic experiments with other values of $\delta$ are planned for future work. Eventually, we trained the final system $UHDS\_doc2vec$ on a concatenation of the reduced general domain corpus $FilteredCorpus$ and the in-domain data $\mathcal{I}n$. Two separate language models were trained with the in-domain data $In$ and the full general domain corpus $Gen$. They have been interpolated and the interpolated model has been used in both $UHBS\_lmi$ (strong baseline) and $UHDS\_doc2vec$ (the submission to the competition). In Figure 1 the pseudocode for filtering the general domain corpus is presented.

Doc2vec filtering selects in one step all the general domain sentences similar to the in-domain producing one $FilteredCorpus$. Eventually, each sentence from $\mathcal{G}en$ is either discarded or added to $FilteredCorpus$).

In order to be able to compare our method with other data selection approaches, we modified the binary decision from step 10 of the algorithm with a step that produces a score for each sentence $s_i \in \mathcal{G}en$ (Figure 2). Therefore, in addition to the submitted systems to the WMT competition, we also conducted experiments with the extended Doc2vec algorithm and with a perplexity-based metric which defines the state-of-the-art for data selection for MT (Axelrod et. al, 2011). We name $SEF$ (Sentence Embedding Filtering) the method presented in Figure 2 and $PPL$ (Perplexity) the state-of-the-art method.

In addition to the input parameters that the algorithm presented in Figure 1 uses, the adapted algorithm receives as input also a percentage $\mathcal{P}$ which gives the number of sentences to be selected from $\mathcal{G}en$. Given a sentence $s_i \in \mathcal{G}en$, the $SEF$ method uses the similarity score between $s_i$ and its $\mathcal{N}$ most similar sentences for producing a final score. Moreover, since the position in $Sim_{s_i}$ matters, we multiply each intermediary score with the inverse position $(\mathcal{N} - j + 1)$. For example, if the most similar sentence to $s_i$ is $s_j$ placed on the first position in $Sim_{s_i}$, then their $score_{ij}$ is multiplied with the highest possible value $\mathcal{N}$. After scoring all the sentences from $\mathcal{G}en$, they are sorted by their score in descending order.

The comparison between $SEF$ and $PPL$ was evaluated on a range of percentages from 10 till 90, incrementing the ratio in steps of 10.

## 5 Results

In this section we present the evaluation scores obtained in the WMT competition for the three sub-

**Algorithm 2** Doc2vec Filtering using percentage $\mathcal{P}$

**procedure** FILTER-PERCENTAGE($\mathcal{I}n, \mathcal{G}en, \mathcal{N}, \delta, \mathcal{P}$)
  $\mathcal{C} \leftarrow \mathcal{I}n + \mathcal{G}en$
  **for each** sentence $s_i \in \mathcal{C}_{source}$ **do**
    tag $s_i$ with the line number $i$
  build vocabulary from tagged $\mathcal{C}_{source}$
  train doc2vec model $\mathcal{M}$ using tagged $\mathcal{C}_{source}$
  **for each** sentence pair $(s_i, t_i) \in \mathcal{G}en$ **do**
    $\mathcal{R}_i = top(\mathcal{N}, mostSimilar(\mathcal{M}, s_i))$
    $Sim_{s_i} = \{(index, score) \in \mathcal{R}_i |\ index \in [1, size_C], score \in (0, 1)\}$
    **for** $(index_j, score_j) \in Sim_{s_i}$ **do**

$$score_{i,j} = \begin{cases} score_{i,j} * (\mathcal{N} - j + 1)^2, & \text{if } index_j < size_{In} \text{ and } score_j > \delta \\ 0, & \text{otherwise} \end{cases}$$

    $$score_i = \sum_{j=1}^{\mathcal{N}} score_{i,j}$$

  sort sentences $\in \mathcal{G}en$ by their score in descending order
  **while** $i \leq \mathcal{P}$ **do**
    add $(s_i, t_i)$ to $FilteredCorpus_{\mathcal{P}}$

Figure 2: Doc2vec filtering algorithm adapted to select a given percentage $\mathcal{P}$ of sentences

mitted systems. Moreover, we present the evaluation scores for the $SEF$ and $PPL$ methods and discuss the results. Table 2 presents the BLEU (Papineni et al., 2002), the BLEU-cased and the TER (Snover et al., 2006) scores for the submitted systems to WMT:

| System | BLEU | BLEU-c | TER |
|---|---|---|---|
| $UHBS\_lmi$ | 37.21 | 35.29 | 0.545 |
| $UHDS\_doc2vec$ | 37.14 | 35.04 | 0.528 |
| $UHBS\_simple$ | 36.02 | 34.17 | 0.546 |

Table 2: Submitted systems results

According to their BLEU scores, the strong baseline, $UHBS\_lmi$, performs almost on a par with the filtered general domain system, $UHDS\_doc2vec$, but with respect to TER $UHDS\_doc2vec$ clearly outperforms the baseline. The results are encouraging, since our selection method filtered out more than 50% of the general domain data without a substantial loss of translation quality compared to the strong baseline.

The BLEU and TER scores for the $SEF$ and $PPL$ methods are given in Table 3. The maximum BLEU score has been achieved by $SEF$ (37.12) selecting 70% of $\mathcal{G}en$. The $PPL$ method achieved its maximum BLEU score at a 90% ratio of $\mathcal{G}en$ with a score of 36.75 that is close to the score already achieved at 30% filtering (36.71). With respect to that, the $SEF$ method also has a close score to it at 30% filtering (36.65). The TER scores are all very close for most of the steps, with the lowest score achieved by the $PPL$ method at 30% filtering (0.532). A very similar score has been gained by the $SEF$ method when filtering to 50% (0.535). In comparison to the systems submitted to WMT, the best BLEU and TER scores have still been achieved by $UHDS\_doc2vec$ and $UHBS\_lmi$.

## 6 Conclusions

In this paper we presented the system the University of Hamburg submitted to the WMT shared task of translating IT texts. We introduced a new method of data selection for filtering the general domain data by searching for sentences that are similar to the in-domain. The novel contribution of our approach consists in using paragraph vectors to capture crucial meaning aspects of a sentence and deploy them to determine intersentential similarity. With less than 50% general domain data the system performs almost as good

| Percentage $\mathcal{P}$ of Gen | **BLEU** | | **TER** | |
|---|---|---|---|---|
| | $\mathcal{SEF}$ | $\mathcal{PPL}$ | $\mathcal{SEF}$ | $\mathcal{PPL}$ |
| 10 | 35.37 | 36.28 | 0.549 | 0.537 |
| 20 | 36.25 | 36.36 | 0.549 | 0.539 |
| 30 | 36.65 | 36.71 | 0.539 | **0.532** |
| 40 | 35.94 | 36.69 | 0.546 | 0.535 |
| 50 | 36.97 | 36.39 | **0.535** | 0.541 |
| 60 | 37.08 | 36.57 | 0.535 | 0.536 |
| 70 | **37.12** | 36.29 | 0.536 | 0.542 |
| 80 | 37.09 | 36.45 | 0.538 | 0.541 |
| 90 | 36.43 | **36.75** | 0.546 | 0.546 |

Table 3: Evaluation results for $SEF$ and $PPL$

as the strong baseline in terms of BLEU.

We also presented an adaptation of the paragraph vector filtering method that is able to select any required percentage of the general domain data and we conducted experiments using a range of ratios for this method and a state-of-the-art method. The BLEU results indicated that the adapted paragraph vector method outperforms the state-of-the-art method.

These results make filtering using paragraph vector for scoring sentences particularly attractive for scenarios where a large pool of general domain data is available, but only a very small amount of in-domain data.

## References


Amittai Axelrod, Xiaodong He and Jianfeng Gao. 2011. Domain Adaptation via Pseudo In-Domain Data Selection. *Proceedings of EMNLP 2011*.

Boxing Chen, Roland Kuhn and George Foster. 2013. Vector Space Model for Adaptation in Statistical Machine Translation *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1285-1293, Sofia, Bulgaria, August 4-9 2013.

George Foster, Cyril Goutte and Roland Kuhn. 2010. Discriminative instance weighting fordomain adaptation in statistical machine translation. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA,USA, pages 451-459.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. *Proceedings of Proceedings of the Sixth Workshop on Statistical Machine Translation*, Edinburgh, UK.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the Translation Model for Statistical Machine Translation based on Information Retrieval. *Proceedings of EAMT 2005*.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for N-gram language modeling. *Proceedings ICASSP*, pages 181-184.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. June 25-27, 2007, Prague, Czech Republic.

Philipp Koehn. 2010. An experimental management system. *The Prague Bulletin of Mathematical Linguistics*, 94.

Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. *Proceedings of the 31st International Conference on Machine Learning*, volume 32, Beijing, China. JMLR: W&CP.

Yajuan Lü, Jin Huang and Qun Liu. 2007. Improving Statistical Machine Translation Performance by Training Data Selection and Optimization. *Prooceedings of EMNLP-CoNLL 2007*.

A. Mandal, D. Vergyri, W. Wang, J. Zheng, A. Stolcke, G. Tur, D. Hakkani-Tür, and N. F. Ayan. 2008. Efficient data selection for machine translation. *Proceedings IEEE Workshop on Spoken Language Technology*.

Saab Mansour, Joern Wuebker and Hermann Ney. 2011. Combining Translation and Language Model Scoring for Domain-Specific Data Filtering. *Proceedings of IWSLT*.

Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160-167, July 07-12, 2003, Sapporo, Japan.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, volume 29, number 1, pages 19-51.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. *Proceedings of*

*the 40th Annual Meeting on Association for Computational Linguistics*, July 07-12, 2002, Philadelphia, Pennsylvania.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45-50.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas*, Cambridge, Massachusetts.

Andreas Stolcke. 2002. SRILM - an Extensible Language Modeling Toolkit. *International Conference on Spoken Language Processing*.

Aleš Tamchyna, Galuščáková Petra, Kamran Amir, Stanojević Miloš and Bojar Ondřej. 2012. Selecting Data for English-to-Czech Machine Translation. *Proceedings of the Seventh Workshop on Statistical Machine Translation*.

Antonio Toral, Pavel Pecina, Longyue Wang and Josef van Genabith. 2015. Linguistically-augmented perplexity-based data selection for language models. *Computer Speech & Language*, Volume 32, Issue 1, July 2015, pages 11-26.

Longyue Wang, Derek F. Wong, Lidia S. Chao, Yi Lu, and Junwen Xing. 2013a. A Systematic Comparison of Data Selection Criteria for SMT Domain Adaptation. *The Scientific World Journal*, vol. 2014.

Longyue Wang, Derek F. Wong, Lidia S. Chao, Junwen Xing and Yi Lu. 2013b. Edit Distance: A New Data Selection Criterion for Domain Adaptation in SMT. *Proceedings of Recent Advances in Natural Language Processing*.

Longyue Wang, Derek F. Wong, Lidia S. Chao, Yi Lu, and Junwen Xing. 2013c. iCPE: A Hybrid Data Selection Model for SMT Domain Adaptation. *Lecture Notes in Artificial Intelligence (LNAI) Springer series*. 12th CCL&1st NLP-NABD