1. Introduction:
Statistical, phrase-based translation is currently the state of the art in machine translation of natural languages. One language pair of interest is Persian-English, which has received little attention until now.

In this report I will present my implementation of such a system, some results regarding its performance, the effect of various data and algorithmic parameters, some lessons learned, and directions for future research.

The rest of this paper is organized as follows: Section 2 offers some background on the problem to highlight its importance. Section 3 is a study of the literature on works related to this topic, while Section 4 describes my methodology and the tools and algorithms that I used. Section 5 describes the two corpora that I worked with to train and test the system. Section 6 has some results pertaining to testing the system under various conditions, and Section 7 discusses some potential future work which is likely to improve on the set of results from this paper. Finally, Section 8 concludes the paper.

2. Background and Motivation:
For example, in the NIST OpenMT machine translation evaluation contest in 2012, several teams participated, offering systems for translating between different language pairs. Persian was the lowest-scoring on average, obtaining only about 19 BLEU points. In contrast, Arabic, which happens to share many aspects including script and vocabulary with Persian, scored 40 points on average. Furthermore, Holzer et. al [Holzer’11], through a very limited analysis (only 5 test sentences), finds the BLEU score of Google Translate for English-Persian to be around 23.5, in contrast to English-French, for example, which was found to be 91. All these show the room for improvement which is present in English-Persian machine translation.

The Persian language is a branch of Indo-European languages, written from right to left with the Arabic alphabet. In contrast to English which is an SVO (subject-verb-object) language, Persian is SOV, which has implications on the levels of word reordering necessary for translation. Furthermore, the Persian morphology, specially that of the colloquial Persian which is used heavily in this study, has a high amount of word inflection. For example, to say “I saw that” in colloquial Persian, one typically only uses one word.
3. Literature Survey:
A number of works have been conducted in order to build a translator between Persian and English. The Shiraz project [Amtrup ‘99], as well as ParsTran, have proposed systems for a rule-based translation. They perform part-of-speech tagging of the languages, and for a specific domain such as sport or science, they build the their translation system. The PeEn-Tran translator [Mohaghegh ‘10 ‘11] is the best statistically trained system reported in the literature, which is very similar to my work in the dataset that it uses and the use of Moses. However, contrary to mine, it trains an English to Persian translator, and focuses on the effect of varying language models, which have come from different Persian corpora such as Iranian newspapers or the BBC Persian service. The best reported BLEU score in this work has been around 31.

4. Data description:
Finding good parallel corpora for Persian-English is a challenging task. Very few publicly available corpora of this type exist, and often times they are of very small sizes or not very clean.
The main dataset that I used in training and testing my system has been the Tehran English Persian (TEP) dataset, which is a collection of around 21,000 subtitle files. The dataset is the largest freely available English-Persian parallel corpus. It consists of 600,000 aligned sentence pairs. Due to the nature of the corpus source, it has heavy usage of colloquial Persian and named entities.

5. Method/Algorithm/Tools:
In this section, I will give a step-by-step description of the algorithms and tools that I used.

5.1 Data pre-processing:
The first step is to clean the data corpus. Due to the nature of community contributions, I found both incorrectness and inconsistencies in the way that different subtitle translations had been written in the Persian language. For this purpose, I wrote some text processing Python scripts to fix the problems that I could identify. This is a best effort attempt and covers as much as possible. It is only to ensure making the most use of the dataset at hand, without having to deal with erroneous training or having to throw away much of the data -- there's already a shortage of that. Below is a list of some text preprocessing that I performed:
1) English characters in the Persian text: For example, the sentence "i am mr. sweeney todd of the fleet street" is translated as "می‌بینم سوئینی تاد هستم از خیابان Fleet". In order to have a clean Persian corpus, I removed these types of sentences.
2) English numbers in the Persian text: A lot of times I noticed that English numbers are inserted verbatim in the Farsi text. This was problematic in two ways: First, it again pollutes the Persian text with English characters and creates inconsistencies, and second, in the right-to-left alignment of the Persian text, the English numbers would be converted right-to-left, creating incorrect numbers. Therefore, I used Python regular expressions to search for all English numbers in the Persian text, reverse the order of digits, and convert them to their corresponding Persian digits.

3) Aligned sentences with significant difference in length between the corpora: Some sentences are translated freely, meaning they don't preserve a close enough word-by-word similarity. One good way to find these sentences is to compare their lengths: if there is a significant difference in their number of words, then there is a good chance that word alignment would not work well on those sentences. One example is the sentence below, which has 6 words and its translation which has 14:

You on a high now love ….

For this reason, I opted to remove sentences with more than 6 words of difference or with more than 2x difference in word number.

4) Some special characters are inserted in the translations which are not helpful for training the system. For example, in one movie, due to erroneous encoding, all single apostrophes were converted to the character. Or similarly, some movies insert a dash before every dialogue, which may or may not be preserved in the translation. Therefore I opted to remove these characters for better matching.

5) Finally, since Persian does not use many English abbreviation signs, it's better to expand them out. Examples of this are the $ and % signs, which I expanded to "dollars" and "percent" respectively.

After all of this data pre-processing is done, the size of the corpus is reduced from 612,000 sentence pairs to 603,000 sentence pairs, which is quite acceptable.

5.2 Tokenization:

Next, the tokens present in each word will be separated individually by blank space. At this stage, the punctuations, apostrophes, etc. will be separated from the words. This is a language-sensitive part of training, and it's important to separate tokens based on the morphemes in each language to get the best mapping between token. Lacking a Persian tokenizer, I used the same English tokenizer for both the source and target. This step will inevitably introduce errors into the Persian side.

5.3 Train/Test/Validate Data Sets:

In building a good statistical tool, it's important to both tune the parameters for quality after training, and to test the performance on a separate dataset. Because I only really had
access to one substantial dataset, I divided it into 3 portions: Training, Validating and Testing.
My experiments were conducted using different ratios of these, as I will discuss in the following sections.
The test set is the truly held-out set, and is only used in the end to measure the performance of the system. In addition to the test set obtained from the TEP parallel corpus, I also used the TED corpus as another small held-out test set.
The language model was trained on the combination of training and validation data, because we have access to those in the training phase.

5.4 Training:
Because my translation system is considering Persian as the source and English as the target language, the language model is trained on the English text. IRSTLM was used as the tool for building the translation model, however since the version of Moses that I used was built to work with the KenLM language model format, I first trained a 3-gram LM using IRSTLM, then converted a textual ARPA format of it with KenLM, and finally binarized the result for faster loading by the decoder.

Next, to train the translation model, the first step is to do word alignment. The Giza++ tool is suitable for this. Employing a heuristic method based on the IBM models, it generates the best guess about which words are translations of each other. Finally, the core of the procedure is implemented with the Moses SMT tool, which considering the possible alignments of words, creates a phrase table and a reordering table to specify the probabilities of different phrases (sequences of tokens) mapping to each other and the probabilities of them being reordered. This is the core model, and several parameters are trained with this. They are as follows:
- Translation score: measure of the adequacy of the translation
- Language model score: measure of the fluency of the translation
- Word penalty: measure of the penalization of too many or too few words in the translation
- Distortion penalty: measure of the penalization of reordering words in the translation
- Unknown word penalty: measure of the penalization of having an out-of-vocabulary word

As a first effort, these parameters are based only on training data. However, in order to improve the quality of the model to work with inputs other than those found in the training set, it is necessary to use the validation set as a tuning input, as described next.

5.5 Tuning
Tuning is done using the Minimum Error Rate Training (MERT) tool. This works using several iterations, each time taking the model of the previous iteration as the initial input
and estimating a better set of parameters based on the validation set. The iterations stop once the change of parameters goes below a certain threshold. This step is crucial to getting higher quality models, however as I show later, there are significant variations in its results and the time it takes to complete based on the choice of the validation set.

5.6 Scoring
Finally, in order to find how well the performance of the model is, I use the held out datasets from both the TEP and the TED parallel corpora, translate their Persian text to English, and use the given English translation to obtain a BLEU score as a metric of how well the system performs.

6. Experiments/metrics:
In this section, I will describe some experiments that I conducted and the findings that came out of them. The hardware to run the training and experiments on turned out to be quite vital, as the algorithms take quite a long time to run on such large datasets. I used a 64-core Intel Xeon 2.3GHz processor, with 128GB of RAM. The times that I report next are from using parallelization capabilities on such hardware.

1) Effect of increasing the test data:
First, to see the effect of holding out more test data, I once got a blue score on a 50,000 sentence pair set and another time on a 5,000 sentence pair set. The figure below shows the variances in the BLEU score. Although the larger test set produced a better score (probably because of canceling out the effect of some unknown words that appeared in the smaller set and affected the end result more fully), the difference seems negligible. However, the larger test sets took about 2 hours to produce a BLEU score while on the small test set this concluded in about 15 minutes. Thus, it seems quite desirable to limit the test set.
2) Effect of large validation set:
As an initial attempt, I divided the corpus up as follows:

*Training: 440,000 sentences*
*Validation: 110,000 sentences*
*Test: 53,145 sentences*

The BLEU score obtained on both the TEP and TED test sets, and during different phases of the tuning process, are depicted in the following figure:

Because the validation set is very large, tuning with MERT takes a very long time. Here, 6 iterations of tuning took 3 days to complete, and it did not converge even then. The following can be observed:

First, domain switching (red vs blue bars) adversely affects the BLEU score, as expected.
Second, there is not much variation during the tuning process, and essentially after two iterations the end result is obtained. Also, tuning tends to make things worse as opposed to the baseline without tuning! This does not occur however in the next experiment, and it might be because of the large validation set which prohibits effective tuning.

3) Effect of smaller validation set:
In the second try, I changed the breakdown of data to train on a significantly larger set, and to tune on a significantly smaller set, as follows:

*Training: 595000 sentence pairs*
*Validation: 5000 sentence pairs*
*Test: 3145 sentence pairs*
This resulted in the whole process of training and tuning to complete in under 7 hours, a significant change from the first attempt. The BLEU score results across 13 iterations (until convergence), is as follows:

![BLEU Score Chart]

The following can be observed:
First, tuning is the bottleneck of training a statistical machine translation system, and it’s important to keep the validation set to a reasonable size.
Second, there is significant increase in the BLEU score in this setting for the in-domain sentences (up to a very good score of around 40 points). However, this is at the cost of very bad out-of-domain scores (down to around 5 points). I conjecture that, although the training set was not polluted by the test set, this breakdown caused a particular movie to appear in both the training and test set, and because the test set was small, this turned out to be significant. For example, if the name of the characters or places in that particular movie are turned from out-of-vocabulary to in-vocabulary, this would have a great impact on the resulting scores. As for the red bars, since the training data was huge compared to the validation data, this ran the risk of overfitting, and it was really specialized to a certain domain.
Third, in this setting, and contrary to the previous experiment, tuning actually does have a positive impact on the BLEU score. I conjecture that because 100,000 sentences is a large amount to tune the parameters on, it cannot be done well and tuning does not actually work well.

7. Future works/directions to recommend:
The most important step in making the English-Persian language pair comparable to other language pairs in terms of the efficacy of Statistical Machine Translation is to collect larger,
cleaner, and more comprehensive parallel corpora. This is an essential step, and several efforts have been made such as using bilingual news websites, but they need to be expanded.

Secondly, a Persian-specific tokenizer might greatly help the training of the model. Moses itself has such capabilities of plugging in new tokenizers, and the developers have welcomed this for more languages. I will making an attempt at this myself in the near future.

Thirdly, more linguistically aware systems may achieve better results, and this capability (using parsers and syntactic information to add to the system) is provided within the “factored translation models” of Moses.

Finally, one work that will have a direct impact, especially for languages with different alphabets than English, is adding transliteration to the training of the model. This would cause out-of-vocabulary words such as named entities, and also numbers, to be transliterated directly into the target language. This could improve the result of my experiments significantly since in the movie domain these words appear quite often. Transliteration systems are usually trained on a letter-by-letter matching basis over the training set [Durrani ‘10].

8. Conclusions:
I have implemented a baseline system to translate from Persian to English. I have studied the tradeoffs of different training/validation/test sets on both the accuracy performance and the timing performance. I also identified a few directions for improving its performance further.

9. References
• F. Och, “Minimum Error Rate Training in Statistical Machine Translation,” ACL 2003
• M. Cettolo et al. “WIT3: Web Inventory of Transcribed and Translated Talks,” EAMT 2012
• M. T. Pilevar et al. “TEP: Tehran English-Persian Parallel Corpus,” CICLing 2012
• Durrani et al. “Hindu to Urdu Machine Translation through Transliteration,” ACL, 2010