ABSTRACT
There are already many existing machine translation (MT) systems but most of them are limited by the paradigms they are using. Different MT systems have different ways of producing a translation for a given source sentence. A language modeler can help improve an MT system’s performance by identifying the best sentence among those produced individually by these engines. There are already existing language modelers for this purpose and one of them is the n-gram language modeler. This model, which was proven to work in [1], suffers from the “curse of dimensionality” wherein word sequences where the model was trained may be different from those seen during evaluation. This means that low probabilities will be assigned to these sequences. A possible solution to address this problem would be to make use of linguistic resources such as WordNet and devise an algorithm such that WordNet features (i.e. synsets and their relations) are combined with the n-grams. By incorporating WordNet features, the language modeler can still give a high score to a word sequence even if there are no exact matches seen in the training corpora. This means that low probabilities will be assigned to these sequences. A possible solution to address this problem would be to make use of linguistic resources such as WordNet and devise an algorithm such that WordNet features (i.e. synsets and their relations) are combined with the n-grams. By incorporating WordNet features, the language modeler can still give a high score to a word sequence even if there are no exact matches seen in the training corpora. This paper presents the proposed solution and its architecture.

Keywords
language modeling, statistical methods, translation quality, WordNet

1. INTRODUCTION

Statistical language modeling (SLM) is used to capture regularities of a natural language. It is employed to improve performance of various natural language processing (NLP) applications. Some of these applications are speech recognition, machine translation, and information retrieval. Machine Translation (MT) translates documents from one language into another. Even though there are many existing MT systems, most of them are limited by the paradigms they are using. Different MT systems have different ways of producing a translation for a given source sentence. A language modeler can help improve an MT system’s performance by identifying the best sentence among those produced individually by these engines.

[1] created a language modeler based on trigram probabilities to choose the best output produced by a number of commercially available MT systems. The best output, in this case, is based on the fluency of a sentence. A similar work is presented in [2] wherein a sentence is formed out of the highest scoring sequence of partial translations. The score is the product of trigram probabilities over the sentence. The problem with the trigram language modelers included in the two previously mentioned works, is that it only gives a high score if a trigram was seen in the training set, this phenomenon is known as the “curse of dimensionality” [3].

There have already been attempts to address this problem. A number of works made use of smoothing techniques such as the work presented in [1] which made use of linear interpolation of trigram, bigram and unigram probabilities. [3] presents a work that combines neural networks and smoothing techniques to the trigram model while the system in [4] made use of additional linguistic features, such as syntax trees, in order to construct a decision tree that will classify sentences.

Another possible solution would be to make use of other linguistic resources such as WordNet and devise an algorithm such that WordNet features (i.e. synsets and their relations) are combined with the n-grams used by approaches in language modelers. By incorporating WordNet features, the language modeler can generalize a sequence which can possibly give it a higher score even if there are no exact matches seen in the training corpora. This may happen when the sequence is seen to be related to the learned data through the aid of WordNet features. [5] already performed a study on the integration of WordNet features with nouns in bigrams. The study used the IS-A relationship of nouns and reported improvement in the language model perplexity although the improvement was below expectation. What can be done is to explore the other semantic relations in nouns and also other syntactic categories (i.e. adjectives, adverbs, verbs) included in WordNet.

2. Conceptual Framework
2.1 WordNet

WordNet is an English lexical dictionary that covers nouns, verbs, adjectives, adverbs and function words. Each syntactic category is organized into synonym sets (synsets), lists of synonymous word forms that are interchangeable in some syntax and have one underlying lexical concept, which are connected by different relation links. Table 1 shows the statistics of WordNet 3.0, the current version of WordNet.
Some of the relations included in WordNet are: synonymy, antonymy, hyponymy/hypernymy and meronymy/holonymy.

Two expressions are said to be synonymous in a linguistic context if their substitution for one for the other does not change the truth value. For example, *plank* and *board* are synonymous over the *carpentry* context because the two words can be used interchangeably in the said context. It must be noted that *board* has another context wherein its synonyms (*e.g.* *committee*) cannot be used to substitute for the *carpentry* context. Synonymy is symmetric such that if *x* is similar to *y*, then *y* is similar to *x*.

The antonym of a word *x* is sometimes not-*x*, but not always. For example, *heavy* and *light* are antonyms, but to say that something is *not heavy* does not imply that it must be *light*. This relationship is the main organizing principle for adjectives and adverbs in WordNet.

The hyponymy/hypernymy relationship is also called the subordination/superordination relationship, subset/superset relationship or IS-A relationship. A concept represented by synset {x₁, x₂, ..., xₙ} is said to be a hyponym of the concept represented by synset {y₁, y₂, ..., yₙ} if there are sentences constructed from such frames as *An x is a (kind of) y*. An example of this relationship is: {maple} is a hyponym of {tree} while *tree* is a hyponym of *plant*. On the other hand, hypernymy is the opposite of hyponymy such that *tree* is the hypernym of *maple* and *plant* is the hypernym of *tree*. This relationship is the main organizing principle for nouns and verbs in WordNet.

The meronymy/holonymy relationship is also called the part-whole relationship or HAS-A relationship. A concept represented by synset {x₁, x₂, ..., xₙ} is said to be a meronym of the concept represented by synset {y₁, y₂, ..., yₙ} if there are sentences constructed from such frames as *A y has an x (as a part) or An x is a part of y*. For example, *finger* is a meronym of *hand*. On the other hand, holonomy is the opposite of meronymy such that *hand* is a holonym of *finger*.

### 2.2 N-Gram Language Model

The n-gram technique models language as a Markov source of order n-1 which means that the probability of each word depends on the previous n-1 words. For example, the probability of a word *wᵢ* depends on *wᵢ₋ₙ₊₁, ..., wᵢ₋₁*. This is represented by:

\[ P(wᵢ|hᵢ) ≈ P(wᵢ|wᵢ₋ₙ₋₁, ..., wᵢ₋₁) \]

The language modeler presented in [1] consists of two phases. The training phase learns the trigrams, bigrams and unigrams and their corresponding frequency counts from the input corpora. The other phase is responsible for evaluating the fluency of sentences based on the results from training.

#### 2.2.1 Training Phase

The training phase accepts raw input corpora and extracts from them trigrams, bigrams and unigrams. The word sequences, together with their frequency count, are stored in a database. Table 2 shows the contents of the database after training the system with the sentence “The big frog ate the big fly”.

### Table 1. WordNet 3.0 Statistics

| Category | Unique Strings | Synsets | Total Word-Sense Pairs |
|----------|----------------|---------|------------------------|
| Noun     | 117097         | 81426   | 145104                 |
| Verb     | 11488          | 13650   | 24890                  |
| Adjective| 22141          | 18877   | 31302                  |
| Adverb   | 4601           | 3644    | 5720                   |
| Totals   | 155327         | 117597  | 207016                 |

### Table 2. Sample Results for the Training Algorithm in Callison-Burch et al. (2001)

| Word Sequence | Frequency Count |
|---------------|-----------------|
| Trigrams      |                 |
| Start-of-sentence | start-of-sentence | 1               |
| The            |                 |
| Start-of-sentence | The big        | 1               |
| The big frog   |                 |
| Big frog ate   |                 |
| Ate the big    |                 |
| The big fly    |                 |
| Big fly end-of-sentence | 1               |
| fly end-of-sentence | end-of-sentence | 1               |
| Big fly        |                 |
| Unigrams       |                 |
| The            | 2               |
| Big            | 2               |
| Frog           | 1               |
| Ate            | 1               |
| Fly            | 1               |

Below is the formula used to compute the probability of a sequence *xyz*.

\[
P(xyz) = \frac{(0.80 \times \text{frequency count of } xyz)}{(\text{frequency count of } xy) + (0.099 \times \text{frequency count of } yz) + (0.14 \times \text{frequency count of } z) + 0.001}
\]
The formula does not only consider the trigram probabilities of word sequences. It includes smoothing of the trigram probabilities by using bigram and unigram probabilities. This is done in order to counteract the effects of sparseness of data. For example, if the trigram “the big bully” was not seen in the training, a $P(\text{xyz}) = 0$ would be computed. However, if smoothing is to be included, there is a possibility of getting a higher score if “big bully” and/or “bully” was seen in the training. As can be seen in the formula, a higher weight is given to trigram occurrences in the training rather than the bigrams and unigrams.

For example, the probability of the sentence “The big frog ate.” will be computed as:

$$P(\text{"The big frog ate"}) = P(\text{"The" start-of-sentence}) \times P(\text{"big" start-of-sentence \text{"The"}}) \times P(\text{"frog" \text{\"The" \text{"big"}}}) \times P(\text{"ate" \text{\"big" \text{"frog"}}}) \times P(\text{end-of-sentence \"frog" \"ate"}) \times P(\text{end-of-sentence \"ate" end-of-sentence})$$

Each of the multiplicands would be computed using the formula for $P(z|xy)$ presented above.

### 2.2.3 Issues

The main problem with the current model is dimensionality. Usually, a word sequence where the language model is used is different from the word sequences used during training. This will then result to trigrams having low probability scores.

For example, the probability of the given sentence “The big insect ate.”, will be computed as follows:

$$P(\text{"The big insect ate"}) = P(\text{"The" start-of-sentence}) \times P(\text{"big" start-of-sentence \text{"The"}}) \times P(\text{"insect" \"The" \text{"big"}}) \times P(\text{"ate" \text{\"big" \text{"insect"}}}) \times P(\text{end-of-sentence \"insect" \"ate"}) \times P(\text{end-of-sentence \"ate" end-of-sentence})$$

Assuming that the data in Table 2 are the only contents of the database, the third to fifth trigrams will have low probabilities. For example, the probability that will be assigned to the third trigram “The big insect” will be low because the sequences “the big insect” and “big insect” are not seen in the database. This means that since only the unigram “fly” is present in the database, the probability of the whole sequence will depend on it such that:

$$P(z|xy) = P(\text{"fly" \"the" \"large"}) \times (0.099 \times \text{frequency count of \"fly\") / (total \# of words seen) + 0.001 \times (0.099 \times 1) / 7 + 0.001 = 0.0151$$

---

3. Proposed Solution and Architecture

As mentioned in Section 1, one possible solution in addressing the “curse of dimensionality” is to incorporate WordNet features to the N-Gram language modeler. Figure 1 shows the architecture of the proposed solution, which is basically the same with the process in the original modeler except that WordNet is added as a resource and additional processing steps are added for the evaluation of sentences.

### 3.1 Training Module

This module handles the extraction of trigrams from a given training corpus. The input corpus, which must be tagged with part of speech (POS), will go through the Sentence Segmentation step to be divided into sentences. These sentences will then be individually processed in the Lexical Trigram and Count Extraction step. This step divides a sentence into trigrams, bigrams and unigrams. These sequences, together with their counts (number of times the sequence is seen during training) and POS tags, are then stored in the Lexical Trigram and Count DB.

### 3.2 Language Evaluation Module

This module handles the selection of the most fluent sentence from a set of pre-tagged (part of speech) candidate sentence translations. The Sentence Assessment Module will process each candidate from the set individually. Here, the sentence passes through the Lexical Trigram Extraction step wherein it will be divided into trigrams. Each trigram will then be matched to the Lexical Trigram and Count DB in the Lexical Trigram Lookup step. When there are no exact matches found in the database, WordNet will be used to get possible matches. Possible matches are trigrams having words that are equal, similar or related to corresponding words (i.e. words in the same position) in the candidate trigram. Since it is possible that there can be more than one match in the database, the system will compute the Possible Lexical Trigram Probability Set (PLTPS).
for the probabilities of each trigram match. Each PLTPS will then be rank from the trigram matches from lowest to highest and then compute for the score of the entire sentence. Finally, given the scores for each candidate translation, the Rank Sentence Score will choose the translation with the highest score.

Going back to the example presented in Section 2.2.3, it is now possible for the unmatched trigrams to have higher probabilities assigned to them. For example, even if the trigram “the big insect” was not seen in the database, WordNet can be used in order to still get a possible match from the database such as “the big fly”. According to WordNet, the words “fly” and “insect” are related through the IS-A relationship. Since these words have some degree of similarity, it would be of an advantage to make use of a fraction of the probability of “the big fly”, the one seen in the training data. One possible way of manipulating the formula would be to factor in the similarity scores such as shown below:

Let:

\[ \text{xyz} = \text{“the large insect”} \]
\[ \text{xyz’} = \text{“the big fly”} \]

Assumption:

\[ \text{similarity-score(“fly”, “insect”) = 0.8} \]

\[ P(z|xy) = P(\text{“insect” | “the” “big”}) \]
\[ = \text{similarity-score(“fly”, “insect”) * P(z’|xy)} \]
\[ = 0.8 * 0.5551 \]
\[ = 0.4441 \]

By making use of the probability of an existing trigram, a higher score, compared to the original probability 0.0151, is assigned to the trigram “The big insect”. By making use of the information in WordNet, it is possible to address the “curse of dimensionality” as seen in this example wherein the trigram “The big insect” was not seen in the training data but is still given a higher probability.

3.3 Resources

There are two resources that are identified to be needed in the architecture: Lexical Trigram and Count DB and WordNet. The Lexical Trigram and Count DB contains the trigrams, together with their corresponding POS tags and counts, learned during training, which is extracted from a corpus. The synsets of WordNet will be used in the Sentence Assessment Module. WordNet is an important resource that will serve as the basis for generalization of trigram matching during cases wherein exact matching is not possible.

4. Evaluation

The two language modelers, the trigram language modeler and the trigram language modeler using WordNet features, will both be evaluated. This will be done in order to compare the language models that will result from the two modelers. This means that the training and test corpora that will be used will be the same for both language modelers.

The language modelers will be evaluated in the English language. The training and test corpora will be in English. The sentences that will be part of the test corpora will be the Filipino-to-English translation results of the three MT engines of the Hybrid English-Filipino Machine Translation project. All parallel translations are assumed to express the same thought.

Linguistic experts shall be called to evaluate the fluency of the parallel outputs of the existing MT systems in the Hybrid English-Filipino Machine Translation project. A scale of 1 to 4 will be used by the linguists to evaluate the sentences. The criteria for evaluation will be explained to them and a practice session will be held for them to comment on the criteria. Afterwards the actual rating would be performed. After the evaluation of the linguists, each set of parallel translations would have a highest scorer (most fluent). These will then be compared to the ones chosen by the language modeler in order to get the percentage of the linguists and the language modeler having chosen the same highest scoring translation.

5. Applications

This research can be used to automatically rate the fluency of sentences. A practical application would be to use this to automatically evaluate natural language generation (NLG) systems such as text simplifiers, text summarizers and story generators. Aside from that, this can also be used to automatically evaluate human-generated sentences especially those of students. In addition, this can be used for the hybridization of MT systems like the Hybrid English-Filipino Machine Translation project. The project, which consists of three parallel working MT engines, can be integrated by a module that will select the best translation among the outputs produced by the three parallel engines. Although the assumption that comes with the research in this application is that the parallel outputs should express the same thought.

6. CONCLUSION

This paper presented the idea of a language modeler that aims to allow the hybridization of MT systems. In addition, the proposed language modeler attempts to address the “curse of dimensionality” by incorporating WordNet features (i.e. synsets and their relations) to n-grams, the language modeler can still give a high score to a word sequence even if there are no exact matches seen in the training corpora.

7. REFERENCES

[1] C. Callison-Burch and R. Flournoy. A program for automatically selecting the best output from multiple machine translation engines. In Proceedings of the Machine Translation Summit VIII, 2001.

[2] J. Carbonell, R. Cohen, A. Lavie, L. Levin, A. Llitjós, E. Peterson, K. Probst, R. Reynolds and S. Vogel. Experiments with a Hindi-to-English Transfer-Based MT System Under a Miserly Data Scenario. In Transactions on Asian Language Information Processing, Special Issue on Rapid Deployment Hindi-English Translation Systems, 2004.

[3] Y. Bengio, R. Ducharme, P. Vincent and C. Jauvin. A Neural Probabilistic Language Model. In Journal of Machine Learning Research 3, 2003.

[4] C. Brockett, S. Corston-Oliver and M. Gamon. A Machine Learning Approach to the Automatic Evaluation of Machine Translation. In Proceedings of the 39th Annual
Meeting on Association for Computational Linguistics, 2001.

[5] R. Hoberman and R. Rosenfeld. Using WordNet to Supplement Corpus Statistics [Online Document]. Available: http://www.cs.cmu.edu/~roseh/Papers/wordnet.pdf, 2002. (March 15, 2007).

[6] R. Beckwith, C. Fellbaum, D. Gross, G. Miller and K. Miller. Five Papers on WordNet [Online Document]. Available: http://wordnet.princeton.edu/5papers.pdf, 1993 (May 18, 2007).