Incorporation of WordNet Features to *n*-gram Features in a Language Modeler *

Kathleen L. Go[a] and Solomon L. See[a]

[a] De La Salle University – Manila
2401 Taft Avenue, Malate
1004 Manila, Philippines
go.kathleen@gmail.com, sees@dlsu.edu.ph

Abstract. *n*-gram language modeling is a popular technique used to improve performance of various NLP applications. However, it still faces the "curse of dimensionality" issue wherein word sequences on which the model will be tested are likely to be different from those seen during training (Bengio *et al.*, 2003). An approach that incorporates WordNet to a trigram language modeler has been developed to address this issue. WordNet was used to generate proxy trigrams that may be used to reinforce the fluency of the given trigrams. Evaluation results reported a significant decrease in model perplexity showing that the new method, evaluated using the English language in the business news domain, is capable of addressing the issue. The modeler was also used as a tool to rank parallel translations produced by multiple Machine Translation systems. Results showed a 6-7% improvement over the base approach (Callison-Burch and Flournoy, 2001) in correctly ranking parallel translations.

Keywords: language modeling, statistical methods, translation quality, WordNet

1. Introduction

n-gram language modeling is a popular statistical language modeling (SLM) technique used to improve performance of various natural language processing (NLP) applications such as speech recognition (SR), machine translation (MT), and information retrieval (IR). However, it still faces the "curse of dimensionality" issue wherein word sequences (i.e. *n*-grams) on which the model will be tested are likely to be different from those seen during training (Bengio *et al.*, 2003). This means that these sequences would always be assigned low probabilities.

There have already been attempts to address this problem. A number of works made use of smoothing techniques such as the one presented in (Callison-Burch and Flournoy, 2001) which made use of linear interpolation of trigram, bigram and unigram probabilities. (Bengio *et al.*, 2003) presents an approach that combines neural networks and smoothing techniques to a trigram model while the work in (Brockett *et al.*, 2001) made use of additional linguistic features, such as syntax trees, in order to construct a decision tree that will classify sentences.

Another possible solution is to make use of other linguistic resources such as WordNet (Fellbaum et al., 2006) and devise an approach that combines WordNet features (i.e. synsets and their relations) with the *n*-grams used in language modelers. By incorporating WordNet features, the language modeler can generate related sequences which can possibly give it a higher score even if there are no exact matches seen during training. (Hoberman and Rosenfeld, 2002) presents a study on the integration of WordNet features to address the data sparseness of nouns in bigrams. The study covered the IS-A relationship of nouns and reported

improvement in the language model perplexity, although the improvement was below expectation.

This paper presents an extension of the study in (Hoberman and Rosenfeld, 2002). A trigram language modeler has been developed that considers other parts of speech (i.e adjective, adverb, verb) and relationships (i.e. HAS-A, Synonymy/Antonymy), in addition to nouns and the IS-A relationship, to address the "curse of dimensionality" issue. Since there are many existing MT systems with different ways of producing translations, the language modeler was used as a tool to automatically rank parallel translations (i.e. translations produced by multiple MT systems). The ranking of translations was based on the method found in (Callison-Burch and Flournoy, 2001) where sentences are ranked based on their fluency, which are computed using the probabilities of their trigram components. Trigrams were used for the *n*-gram language modeler since the study in (Callison-Burch and Flournoy, 2001) showed that trigrams are effective when used in ranking parallel translations.

Section 2 discusses the architecture of the language modeler while Section 3 presents an example on how the new approach works. Section 4 presents the evaluation results both in terms of addressing the "curse of dimensionality" issue and when applied to the ranking of parallel translations. Lastly, Section 5 contains the conclusions and the discussion of possible future works for this research.

2. Architecture

The language modeler has two modules: the Training Module and the Language Evaluation Module. The latter, in turn, consists of the Base Language Evaluation Submodule and the WordNet-Integrated Language Evaluation Submodule.

2.1.Training Module

The Training Module, shown in Figure 1, is based on the language modeler presented in (Callison-Burch and Flournoy, 2001). This module handles the extraction of trigrams, bigrams, and unigrams from an input document. Since WordNet only contains lemma and general representations of parts of speech (e.g. "noun" instead of "NNP" for proper nouns), the general representations are needed for the evaluation of trigrams.

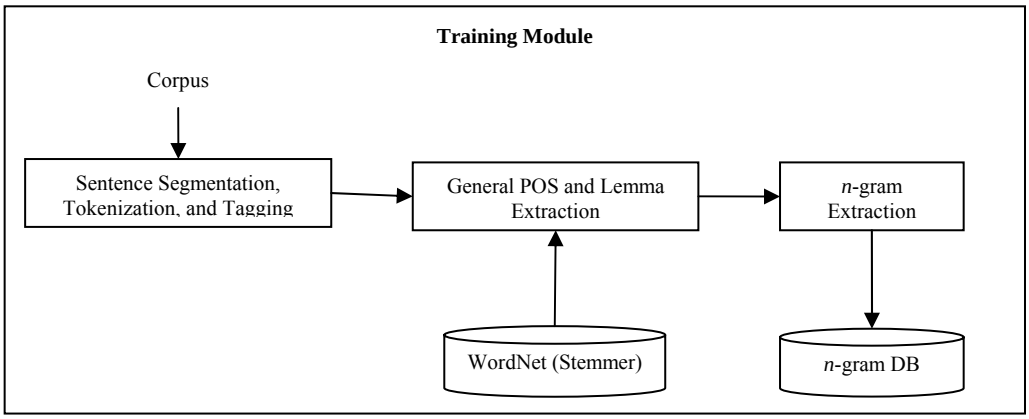

Figure 1: Training Module Architecture.

2.2.Language Evaluation Module

The Language Evaluation Module, shown in Figure 2, consists of the Base Language Evaluation Submodule and the WordNet-Integrated Language Evaluation Submodule.

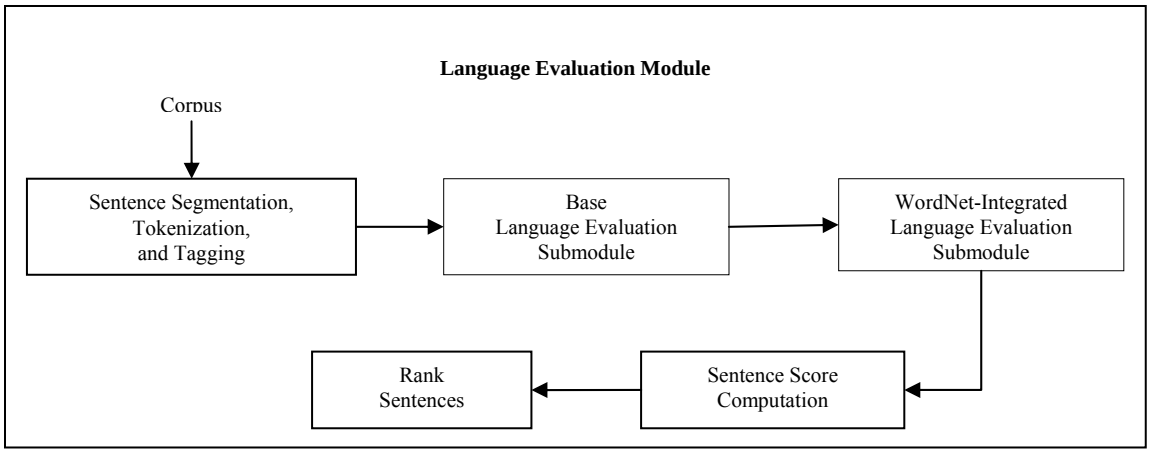

Figure 2: Language Evaluation Module Architecture.

2.2.1. Base Language Evaluation Submodule

The Base Language Evaluation Submodule, shown in Figure 3, is a modified version of the evaluation process in (Callison-Burch and Flournoy, 2001).

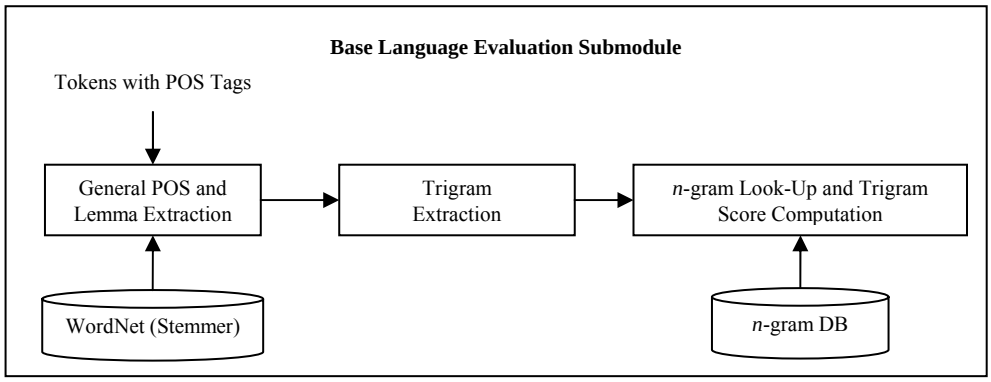

Figure 3: Base Language Evaluation Submodule Architecture.

After the Sentence Segmentation, Tokenization, and Tagging process, the input is passed to the General POS and Lemma Extraction process for the extraction of the generalized representation of its tokens and POS tags. Generalized parts of speech are extracted from the POS tags generated by MontyTagger (Hugo, 2004) while the lemma representations of the tokens are extracted by using WordNet's stemmer. The Trigram Extraction process then extracts the trigrams that will be evaluated, including their POS sequence and general representation. These trigrams are passed to the *n*-gram Look-Up and Trigram Score Computation process that handles the matching of trigrams and their components in the *n*-gram DB to get their frequencies and compute their probabilities. This process performs a *Lemma and General POS Look-Up* whereas the original version performs a *Surface Level without POS Look-Up*. In the former, the *n*-gram Look-Up process matches base on the general representation of the *n*-grams while in the latter, the process matches base only on the surface level (i.e. actual word sequences) of the *n*-grams. The formula used to compute trigram probabilities, which was adapted from (Callison-Burch and Flournoy, 2001), is shown below.

$$P(z|xy) = (0.80 * \text{frequency of } xyz) / (\text{frequency of } xy) + (0.14 * \text{frequency of } yz) / (\text{frequency of } y) + (0.099 * \text{frequency of } z) / (\text{total \# of words seen}) + 0.001 \quad (1)$$

2.2.2. WordNet-Integrated Language Evaluation Submodule

The WordNet-Integrated Language Evaluation Submodule, shown in Figure 4, contains the additional processes that correspond to the new approach. In summary, the process starts with using the contents of WordNet to generate proxy trigrams for the trigrams that were not seen during training. Depending on the presence of the proxy trigrams or their components in the *n*-gram DB, they may be used to reinforce the fluency of the given trigrams. This is done by using the scores of the proxy trigrams to reinforce the score of the given trigram. Originally, only one proxy trigram was used to reinforce trigram scores. However, initial results showed that using only one proxy trigram, either the highest scoring or the most similar, does not give significant increases in trigram scores. Since proxy trigrams are also faced with the "curse of dimensionality" issue, multiple proxy trigrams were used. If a proxy trigram has matches that contain at least one proxy word, it can be used to reinforce the fluency of the given trigram or its lower order components (i.e. bigram and unigram components). This approach also addresses the data sparseness issue of the given trigram's components wherein even if they have matches in the *n*-gram DB, they are still assigned low scores.

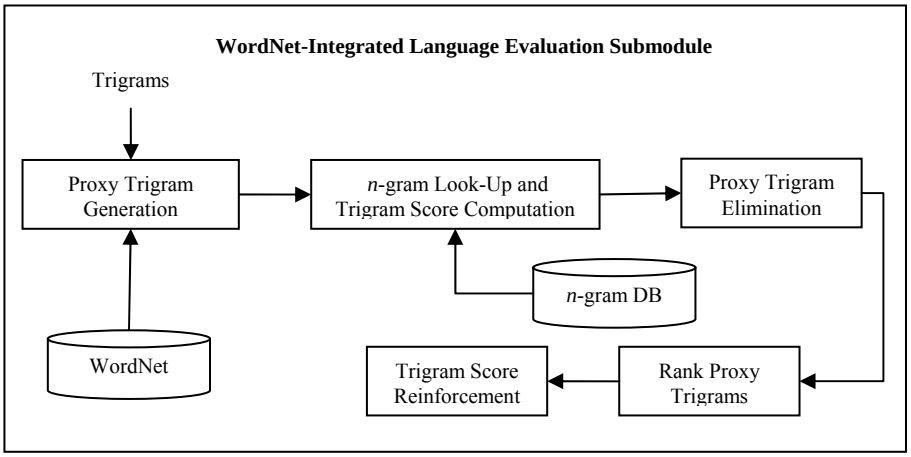

Figure 4: WordNet-Integrated Language Evaluation Submodule Architecture.

In detail, an unseen trigram first goes through the Proxy Trigram Generation process. This process generates proxy trigrams by first looking for partially matching trigrams in the *n*-gram DB (e.g. only the first two words are matched). Words having no position matches (i.e. words that caused the trigrams to not exactly match) are replaced with proxy words (i.e. related words from WordNet). Proxy words are related to the original word through the IS-A and Synonymy/Antonymy relationship. Originally, words related through the HAS-A relationship were also retrieved for nouns. However, initial results showed that the increase in reinforced scores caused by proxy trigrams containing HAS-A related proxy words are insignificant.

The resulting proxy trigrams are then passed to the *n*-gram Look-Up and Trigram Score Computation process, which is the same with the one in the Base Language Evaluation Submodule. In this process, the probabilities of the proxy trigrams are computed.

The Proxy Trigram Elimination process would then remove the proxy trigrams that would not aid in the reinforcement of the given trigram's fluency. These are the proxy trigrams with no matches (i.e. exact, bigram or unigram) in the *n*-gram DB or proxy trigrams with matches but do not contain at least one proxy word. In the initial study, proxy trigrams undergo the Similarity Score Computation process before *n*-gram Look-Up and Trigram Score Computation. In the said process, the similarity between a word and its proxy word is computed using WordNet::Similarity (Banarjee et al., 2006). These scores were used in Proxy Trigram Elimination to further filter the proxy trigrams. This is done by using pre-computed similarity score thresholds for each POS such that if the similarity scores of the proxy words in a proxy trigram do not meet the thresholds, the proxy trigram is assumed to be non-fluent and is

removed. The thresholds were derived from the range of the similarity scores of the proxy words belonging to proxy trigrams that produced fluent proxy sentences (i.e. sentences with proxy trigram replacements). The initial results showed that the thresholds were not effective in its purpose such that only an insignificant amount of proxy trigrams were being filtered out. Therefore, the Similarity Score Computation process and the use of similarity score thresholds are removed.

After the Proxy Trigram Elimination process, the remaining proxy trigrams are passed to the Rank Proxy Trigrams process to be ranked based on the probabilities assigned to them from highest to lowest. The Trigram Score Reinforcement process then makes use of the remaining proxy trigrams to reinforce the fluency of the given trigram. This is done by integrating the original score of the given with the scores of the remaining proxy trigrams. The formula used to compute reinforced trigram scores is shown below.

$$P(xyz)_{\text{reinforced}} = (1-\lambda)P(xyz) + \lambda\left(\sum_{i=1}^{l} P(xyz_{\text{proxyTriMatch}}) + \sum_{i=1}^{m} P(xyz_{\text{proxyBiMatch}}) + \sum_{i=1}^{n} P(xyz_{\text{proxyUniMatch}})\right) \quad (2)$$

where:

$$\lambda = 0.9\,,$$

$$\sum_{i=1}^{l} P(xyz_{\text{proxyTriMatch}}) <= 0.9\,,\ \sum_{i=1}^{m} P(xyz_{\text{proxyBiMatch}}) <= 0.01\,,\ \sum_{i=1}^{n} P(xyz_{\text{proxyUniMatch}}) <= 0.006$$

The purpose of λ is to prevent non-fluent trigrams from being reinforced too much in case it has a proxy trigram that is fluent. If a trigram has proxy trigrams that passed the Proxy Trigram Elimination process, the trigram would most likely have a higher reinforced score. This cannot be prevented even for non-fluent trigrams. Therefore, part of the reinforced trigram score must still be influenced by the original score. The value of λ was derived by getting the value with the highest performance in the ranking of parallel translations.

As seen in the formula, the total proxy score for each trigram is derived by getting the sum of: 1) total score of the proxy trigrams with exact matches, 2) total score of the proxy trigrams with bigram matches, and 3) total score of the proxy trigrams with unigram matches. The total scores for each match level are separate since they are assigned different score limits in order to prevent lower level matches from causing a high increase in the reinforced score. If there is only one threshold that covers all the match levels, the reinforced score of a trigram having many proxy trigrams with lower level matches may equal or exceed the score of a trigram that has a proxy trigram with an exact match. This is possible because of the data sparseness issue wherein even if a trigram or proxy trigram has an exact match, the scores assigned to them may still be low. Also, there is a greater chance of a trigram having proxy trigrams with lower level matches than having proxy trigrams with exact matches. Therefore, in order to prevent the scores of the lower level matches to consume the portion of the score that is supposed to be for higher level matches, different score limits are assigned to each match level. These values were derived through multiple tests conducted wherein different values were used to compute reinforced scores of corresponding fluent and non-fluent sets. The results were compared and the set of threshold values that gave a high perplexity reduction while at the same time maintained the perplexity distance was chosen.

2.2.3. Sentence Score Computation

After the scores and reinforced scores of the trigrams are computed, the Sentence Score Computation process computes the fluency scores of the sentences. The fluency score of a

sentence is determined by getting the product of the probabilities assigned to all its trigram components. If a trigram has a reinforced score, it is used instead of the original score.

2.2.4. Rank Sentences

Each of the sentence sets (i.e. parallel translation sets) are then passed to the Rank Sentences process to be ranked. The ranking of sentences is based on fluency score from highest to lowest.

3. Example

This section presents an example of the new approach. In this example, the given trigram "*to local capitalists*" – "TO JJ NNS" with a general representation of "*to local capitalist*" – "TO adjective noun", only has a unigram match in the *n*-gram DB with a score of 0.008. This section shows how the reinforced trigram score is computed.

In the Proxy Trigram Generation process, the first step is to look for trigrams with partial matches in the *n*-gram DB. In this case, assume that the *n*-gram DB contains two trigrams having partial matches with the given: 1) "*a foreign capitalist*" – "DT JJ NN" ("*a foreign capitalist*" – "DT adjective noun") and 2) "*to local investors*" – "TO JJ NNS" ("*to local investor*" – "TO adjective noun"). But since the second trigram has more position matches compared to the first (i.e. 2 position matches vs. 1 position match), only the second is used as a pattern (i.e. "*to local {x}*"). There can be more than one pattern used as long as they have the same number of matching words (e.g. "*to {x} investors*"). WordNet is then accessed to retrieve words related to the words with no position matches. Table 1 shows a list of some of the proxy words for the word "*capitalists*" including their relationships.

Table 1: List of Sample Proxy Words

Proxy Word	Relationship		Proxy Word	Relationship
conservative	hypernymy		holder	hyponymy
conservativist	hypernymy		materialist	hyponymy
person	hypernymy		businessperson	hyponymy
individual	hypernymy		investor	hyponymy
someone	hypernymy		financier	hyponymy

The proxy words are used to generate proxy trigrams by using the patterns (i.e. "*to local {x}*") and replacing their counterparts. Table 2 shows the generated proxy trigrams for the example.

Table 2: Results of the Proxy Trigram Generation Process

ID	Proxy Trigram	Proxy Trigram POS	Proxy Trigram Lemma	Proxy General POS
1	to local conservative	TO JJ NNS	to local conservative	TO adjective noun
2	to local conservativist	TO JJ NNS	to local conservativist	TO adjective noun
3	to local person	TO JJ NNS	to local person	TO adjective noun
4	to local individual	TO JJ NNS	to local individual	TO adjective noun
5	to local someone	TO JJ NNS	to local someone	TO adjective noun
6	to local holder	TO JJ NNS	to local holder	TO adjective noun
7	to local materialist	TO JJ NNS	to local materialist	TO adjective noun
8	to local financier	TO JJ NNS	to local financier	TO adjective noun
9	to local businessperson	TO JJ NNS	to local businessperson	TO adjective noun
10	to local investor	TO JJ NNS	to local investor	TO adjective noun

The proxy trigrams are then passed to the *n*-gram Look-Up and Trigram Score Computation process. Assume that the *n*-gram DB has contents that produced the results in Table 3.

Table 3: Results of the *n*-gram Look-Up and Trigram Score Computation Process

ID	Proxy Trigram Lemma	Proxy General POS	Score	Match Level
1	to local conservative	TO adjective noun	0.008	Unigram
2	to local conservativist	TO adjective noun	0.001	None
3	to local person	TO adjective noun	0.008	Unigram
4	to local individual	TO adjective noun	0.001	None
5	to local someone	TO adjective noun	0.008	Unigram
6	to local holder	TO adjective noun	0.008	Unigram
7	to local materialist	TO adjective noun	0.001	None
8	to local financier	TO adjective noun	0.001	None
9	to local businessperson	TO adjective noun	0.001	None
10	to local investor	TO adjective noun	0.948	Trigram

Given the results shown in Table 3, the Proxy Trigram Elimination process removes the proxy trigrams that would not be useful in reinforcing the given trigram's fluency. These are the proxy trigrams with no matches in the *n*-gram DB. Also, if the match of a proxy trigram does not contain at least one proxy word, the proxy trigram is eliminated. Table 4 shows the remaining proxy trigrams after the Proxy Trigram Elimination and the Rank Proxy Trigrams processes.

Table 4: Results of the Proxy Trigram Elimination Process

ID	Proxy Trigram Lemma	Proxy General POS	Score	Match Level
10	to local investor	TO adjective noun	0.948	Trigram
1	to local conservative	TO adjective noun	0.008	Unigram
3	to local person	TO adjective noun	0.008	Unigram
5	to local someone	TO adjective noun	0.008	Unigram
6	to local holder	TO adjective noun	0.008	Unigram

The Trigram Score Reinforcement process would then use the remaining proxy trigrams to reinforce the score of the given. The equation for the reinforced trigram score is shown below. The value 0.008 represents the original score of the given trigram. For the total score of the proxy trigrams with trigram matches, the threshold value (i.e. 0.9) was used because the sum (i.e. 0.948) exceeded the threshold value. Since there are no proxy trigrams with a bigram match, the total score is 0. For the total score of the proxy trigrams with unigram matches, the threshold value (i.e. 0.006) was used because the sum (i.e. 0.032) exceeded the threshold value. The reinforced score of the given trigram is 0.8162.

$$P(\text{"}\textit{to local capitalists}\text{"})_{\text{reinforced}} = (0.1 * 0.008) + (0.9)(0.9 + 0 + 0.006) = 0.8162 \qquad (3)$$

4. Evaluation Results

The trigram language modeler has been trained with online articles in the English language and under the business news domain. The training set consists of 1.86 million words while the evaluation set consists of four sets of English sentences, each set having 100 sentences. The four sets of sentences are parallel translations of 100 Filipino sentences selected from online articles, which are also under the business news domain. The first two sets are manual translations while the remaining are translations of the REAL Translation system (Alcantara *et al.*, 2006) and TExt Translation (Go *et al.*, 2006). The first set of manual translations contains

more fluent sentences compared to the other set while almost all of the sentences in the two sets of automatic translations are non-fluent.

The language modeler underwent two evaluation processes: automatic evaluation through the comparison of perplexity values and manual evaluation through comparison of the automatic and manual ranking of the parallel translation sets.

For automatic evaluation, the reduction in the perplexity values caused by the new approach (i.e. WordNet-Integrated Language Evaluation) was measured. Table 5 shows the perplexity reductions resulting from comparisons to the two versions of the Base Language Evaluation: Version 1 uses *Surface Level without POS Look-Up* (Callison-Burch and Flournoy, 2001) while Version 2 uses *Lemma and General POS Look-Up*. The big reductions in perplexity (e.g. 50.25% reduction for Set 1 when compared to the perplexity of Base Language Evaluation Version 1) show that the WordNet-Integrated approach was successful in reinforcing trigram scores. However, a reduction in perplexity does not necessarily mean that there would be a reduction in the error rate for the specific application of the language modeler. Also, since it cannot be avoided that the perplexity of the input be reduced, including those that are not fluent, the distance between the perplexity of the fluent and non-fluent sets must be inspected.

Table 5: Perplexity and % Perplexity Reduction of the WordNet-Integrated Language Evaluation Submodule over the Evaluation Set

	WordNet-Integrated Language Evaluation	
	Perplexity	% Perplexity Reduction
Set 1	44.05073	
Base Language Evaluation Version 1		50.23%
Base Language Evaluation Version 2		45.54%
Set 2	43.97654	
Base Language Evaluation Version 1		52.45%
Base Language Evaluation Version 2		47.91%
Set 3	210.6316	
Base Language Evaluation Version 1		42.27%
Base Language Evaluation Version 2		39.02%
Set 4	262.4471	
Base Language Evaluation Version 1		42.59%
Base Language Evaluation Version 2		37.26%

Table 6 shows that the new approach is consistent in maintaining the distance between the perplexities of Set 3 and Set 4 when compared to Set 1 and Set 2. For example, Set 3 is about 378% more perplexed than Set 1 and Set 2. However, Set 2, which was supposed to contain less fluent sentences, became even less perplexed compared to Set 1. This is because the sentences in Set 2, although less fluent than those in Set 1, contains only a small number of errors that cause them to become less fluent. The decrease in perplexity is caused more by the fluent trigrams.

For manual evaluation, five monolingual English speakers were asked to rank the fluency of the parallel translations in the evaluation set. The same set was also ranked by three language modelers using the different versions of Language Evaluation. The results of the automatic rankings were compared to the results of the manual ranking to get the accuracy of the modelers. Table 7 shows the results of the comparison. Aside from the number of correctly ranked parallel translations, the number of correctly ranked sentences on each set was checked in order to determine which sets the modelers had difficulty in ranking. The WordNet-Integrated Language Evaluation version has 67% accuracy which means that it has a 7% improvement over the Base Language Evaluation using *Surface Level without POS Look-Up*

(Callison-Burch and Flournoy, 2001) and a 6% improvement over the Base Language Evaluation using *Lemma and General POS Look-Up*.

Table 6: Comparison of the Perplexity of the Evaluation Set From the WordNet-Integrated Language Evaluation Submodule

Set #	Set 1	Set 2	Set 3
Set 1			
Set 2	-0.17%		
Set 3	378.16%	378.96%	
Set 4	495.78%	496.79%	24.6%

The results in Table 7 confirmed the results in the automatic evaluation. The WordNet-Integrated approach was able to correctly discriminate fluent sentences from the non-fluent ones (Set 1 and Set 2 vs. Set 3 and Set 4) as seen from the high accuracy rating for Set 3 and Set 4. Another factor for the high accuracy is that according to the manual evaluators, 92% of the parallel sentences belonging to these two sets have the same rank (i.e. same level of fluency). Therefore, for almost all of the parallel translations from the two sets, the language modeler was not required to identify which one is more fluent than the other. However, this is not the case with the translations from Set 1 and Set 2 wherein only 40% have the same fluency level. This means that for 60% of the parallel translations between Set 1 and Set 2, the modelers are required to correctly choose between the translations from the two sets. The accuracy ratings for Set 1 and Set 2 show that the modelers, including the one using the WordNet-Integrated approach, are having difficulty in choosing between sentences with almost the same fluency level (e.g. between the first and second most fluent).

Table 7: Comparison of Manual and Automatic Rankings of the Evaluation Set

Version	# (%) of correctly ranked parallel translations	# (%) of correctly ranked translations per set			
		Set 1	Set 2	Set 3	Set 4
Base Language Evaluation Version 1	60	67	65	93	91
Base Language Evaluation Version 2	61	69	65	93	92
WordNet-Integrated Language Evaluation	67	72	68	95	94

Since the new approach was developed to address the "curse of dimensionality" issue, an additional evaluation was conducted on Set 1 and Set 2 to check if it will be more effective in ranking the translations if there are more unseen trigrams in the input sentences. There are originally 610 trigrams having an exact match from Set 1 and 649 trigrams from Set 2. These trigrams, which make up about 50% of the total number of trigrams in Set 1 and Set 2, were reduced by about 50% more. The language modeler using the WordNet-Integrated Language Evaluation Submodule had 73% accuracy, which is even less than the 74% accuracy of the two versions of Base Language Evaluation.

In order to get the causes for the errors made by the new approach, the 33 incorrectly ranked parallel translations were inspected. We learned that not all causes for the incorrect rankings belong to the scope of the study. For example, less fluent sentences that are shorter were assigned higher ranks than the longer fluent ones. However, there are also instances that must be solved. For example, there are cases wherein higher reinforced trigram scores were assigned to the less fluent trigrams. Another case is when less fluent trigrams are assigned higher reinforced scores than their more fluent and exact matching counterparts.

5. Conclusion and Future Work

A new approach has been developed that integrated WordNet features to an *n*-gram language modeler to address the "curse of dimensionality" issue. It was also able to address the data sparseness issue of lower order components of *n*-grams. Evaluation results show that it is successful in addressing "curse of dimensionality" by reinforcing trigram scores. When used to rank parallel translation sets, it had a 6-7% improvement over the base approaches in terms of accuracy. This new methodology is not effective in choosing between translations with almost the same level of fluency.

One possible extension to this research is to use an ontology specific to the domain the language modeler is intended to be used on. Since WordNet does not contain domain classifications, there are cases wherein less fluent trigrams are assigned high reinforced scores. WordNet contains too many words such that even those that are not related to the sense or domain of the given are still used in the fluency reinforcement process. There is also no way of determining if a proxy trigram is fluent or not since initial results proved that using similarity scores as thresholds is not effective in discriminating between fluent and non-fluent proxy trigrams. Therefore, all of the proxy trigrams with a match may be used in the reinforcement of trigram scores. By using an ontology, the number of applicable proxy trigrams will be reduced and a new set of similarity score thresholds can be derived. These changes may prove to be more effective in approximating the fluency of trigrams.

Another possible extension is to also reinforce the scores of those trigrams already seen in the training. Due to the data sparseness issue in *n*-gram language models, there are cases wherein reinforced trigram scores are higher that their exact matching counterparts.

References

Alcantara, D., B. Hong, A. Perez, and L. Tan. 2006. REAL Translation: Rule Extraction Applied in Language Translation. De La Salle University.

Banarjee, S., J. Michelizzi, S. Patwardhan and T. Pedersen. 2006. WordNet::Similarity 1.04 [online]. Available: http://sourceforge.net/projects/wn-similarity. (May 20, 2007).

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

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

Callison-Burch, C. and R. Flournoy. 2001. A Program for Automatically Selecting the Best Output from Multiple Machine Translation Engines. *Proceedings of the Machine Translation Summit VIII*.

Fellbaum, C., B. Haskell, H. Langone, G. Miller, R. Poddar, R. Tengi and P. Wakefield. 2006. WordNet 2.1 [online]. Available: http://wordnet.princeton.edu/obtain. (May 20, 2007).

Go, K., M. Morga, V. Nuñez, and F. Veto. 2006. Text Translation: Template Extraction for a Bidirectional English-Filipino Example-Based Machine Translation. De La Salle University.

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

Hugo, L. 2004. MontyLingua V.2.1 [online]. Available: http://web.media.mit.edu/~hugo/montylingua/. (January 6, 2008).