Near-Synonym Choice in Natural Language Generation

Diana Zaiu Inkpen and **Graeme Hirst**
Department of Computer Science
University of Toronto
Toronto, ON, Canada, M5S 3GS
{dianaz,gh}@cs.toronto.edu

Abstract

We present Xenon, a natural language generation system capable of distinguishing between near-synonyms. It integrates a near-synonym choice module with an existing sentence realization module. We evaluate Xenon using English and French near-synonyms.

1 Introduction

Natural language generation systems need to choose between *near-synonyms* – words that have the same meaning, but differ in lexical nuances. Choosing the wrong word can convey unwanted connotations, implications, or attitudes. The choice between near-synonyms such as *error*, *mistake*, *blunder*, and *slip* can be made only if knowledge about their differences is available.

In previous work (Inkpen & Hirst 01) we automatically built a lexical knowledge base of near-synonym differences (LKB of NS). The main source of knowledge was a special dictionary of near-synonym discrimination, *Choose the Right Word* (Hayakawa 94). The LKB of NS was later enriched (Inkpen 03) with information extracted from other machine-readable dictionaries, especially the Macquarie dictionary.

In this paper we describe Xenon, a natural language generation system that uses the knowledge of near-synonyms. Xenon integrates a new near-synonym choice module with the sentence realization system named HALogen[1] (Langkilde & Knight 98), (Langkilde-Geary 02b). HALogen is a broad-coverage general-purpose natural language sentence generation system that combines symbolic rules with linguistic information gathered statistically from large text corpora (stored in a language model). For a given input, it generates all the possible English sentences and it ranks them according to the language model, in order to choose the most likely sentence as output.

Figure 1 presents Xenon's architecture. The input is a semantic representation and a set of preferences to be satisfied. A concrete example of input is shown in Figure 2. The final output is a set of sentences and their scores. The first sentence (the highest-ranked) is considered to be the solution.

[1]http://www.isi.edu/licensed-sw/halogen/

2 Meta-concepts

The semantic representation that is one of Xenon's inputs is represented, like the input to HALogen, in an interlingua developed at ISI (Information Science Institute, University of Southern California). As described in (Langkilde-Geary 02b), this language contains a specified set of 40 roles, and the fillers of the roles can be words, concepts from Sensus (Knight & Luk 94), or complex representations.

Xenon extends the representation language by adding meta-concepts. The meta-concepts correspond to the core denotation of the clusters of near-synonyms, which is a disjunction (an OR) of all the senses of the near-synonyms of the cluster.

We use distinct names for the various meta-concepts. The name of a meta-concept is formed by the prefix "generic", followed by the name of the first near-synonym in the cluster, followed by the part-of-speech. For example, if the cluster is *lie, falsehood, fib, prevarication, rationalization, untruth*, the name of the cluster is "generic_lie_n". If there are cases where there could still be conflicts after differentiating by part-of-speech, the name of the second near-synonym is used. For example, "stop" is the name of two verb clusters; therefore, the two clusters are renamed: "generic_stop_arrest_v" and "generic_stop_cease_v".

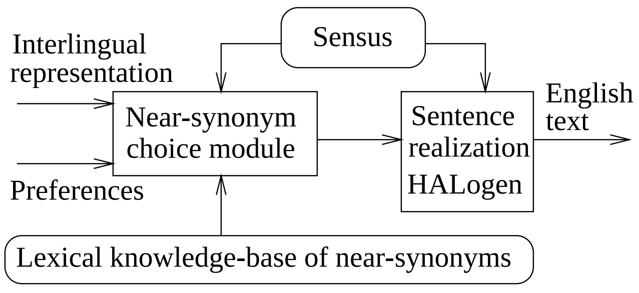
Sensus
Interlingual representation
Preferences
Near-synonym choice module
Sentence realization HALogen
English text
Lexical knowledge-base of near-synonyms

Figure 1: The architecture of Xenon.

Input: (A9 / tell :agent (V9 / boy) :object (O9 / generic_lie_n)
Input preferences:
    ((DISFAVOUR :AGENT)
     (LOW FORMALITY)
     (DENOTE (C1 / TRIVIAL)))
Output:
The boy told fibs. −40.8177
Boy told fibs. −42.3818
Boys told fibs. −42.7857

Figure 2: Example of input and output of Xenon.

3 Near-synonym choice

Near-synonym choice involves two steps: expanding the meta-concepts, and choosing the best near-synonym for each cluster according to the preferences. We implemented this in a straightforward way: the near-synonym choice module computes a satisfaction score for each near-synonym; then satisfaction scores become weights; in the end, HALogen makes the final choice, by combining these weights with the probabilities from its language model. For the example in Figure 2, the expanded representation, which is input to HALogen, is presented in Figure 3. The near-synonym choice module gives higher weight to *fib* because it satisfies the preferences better than the other near-synonyms in the same cluster. Section 4 will explain the algorithm for computing the weights.

4 Preferences

The preferences that are input to Xenon could be given by the user, or they could come from an analysis module if Xenon is used in a machine translation system (corresponding to nuances of near-synonyms in a different language). The formalism for expressing preferences and the preference satisfaction mechanism is adapted from the prototype system I-Saurus (Edmonds & Hirst 02).

The preferences, as well as the distinctions between near-synonyms stored in the LKB of NS, are of three types. Stylistic preferences express a certain formality, force, or concreteness level and have the form: (strength stylistic-feature), for example (low formality). Attitudinal preferences express a favorable, neutral, or pejorative attitude and have the form: (stance entity), where stance takes the values favour, neutral, disfavour. An example is: (disfavour :agent). Denotational preferences connote a particular concept or configuration of concepts and have the form: (indirectness peripheral-concept), where indirectness takes the values suggest, imply, denote. An example is: (imply (C / assessment :MOD (OR ignorant uninformed))). The peripheral concepts are expressed in the ISI interlingua.

In Xenon, preferences are transformed internally into pseudo-distinctions that have the same form as the corresponding type of distinctions. The distinctions correspond to a particular near-synonym, and also have frequencies – except the stylistic distinctions. In this way preferences can be directly compared to distinctions. The pseudo-distinctions corresponding to the previous examples of preferences are:

(– low formality)
(– always high pejorative :agent)
(– always medium implication (C / assessment
        :MOD (OR ignorant uninformed))).

For each near-synonym w in a cluster, a weight is computed by summing the degree to which the near-synonym satisfies each preference from the set P of input preferences:

$$Weight(w,P) = \sum_{p \in P} Sat(p,w).$$

The weights are then transformed through an exponential function that normalizes them to be in the interval $[0,1]$. The exponentials function that we used is:

$$f(x) = \frac{e^{x^k}}{e-1}$$

The main reason this function is exponential is that the differences between final weights of the near-synonyms from a cluster need to be numbers that are comparable with the differences of probabilities from HALogen's language model. The method for choosing the optimal value of k is presented in Section 7.

For a given preference $p \in P$, the degree to which it is satisfied by w is reduced to computing the similarity between each of w's distinctions and a pseudo-distinction $d(p)$ generated from p. The maximum value over i is taken:

$$Sat(p,w) = max_i \; Sim(d(p), d_i(w)),$$

where $d_i(w)$ is the i-th distinction of w. We explain the computation of *Sim* in the next section.

5 Similarity of distinctions

The similarity of two distinctions, or of a distinction and a preference (transformed into a distinction), is computed similarly to (Edmonds 99):

```
(A9 / tell   :agent (V9 / boy)
  :object (OR
     (e1 / (:CAT NN :LEX “lie”) :WEIGHT 1.0e−30)
     (e2 / (:CAT NN :LEX “falsehood”) :WEIGHT 6.93e−8)
     (e3 / (:CAT NN :LEX ”fib”) :WEIGHT 1.0)
     (e4 / (:CAT NN :LEX “prevarication”) :WEIGHT 1e−30)
     (e5 / (:CAT NN :LEX “rationalization”) :WEIGHT 1e−30)
     (e6 / (:CAT NN :LEX “untruth”) :WEIGHT 1.38e−7))
```

Figure 3: The interlingual representation from Fig. 2 after expansion by the near-synonym choice module.

$$Sim(d_1,d_2) = \begin{cases} Sim_{den}(d_1,d_2) \\ Sim_{att}(d_1,d_2) \\ Sim_{sty}(d_1,d_2) \end{cases} \qquad (1)$$

If the two distinctions are of different type, their similarity is zero.

Distinctions are formed out of several components, represented as symbolic values on certain dimensions. In order to compute a numeric score, each symbolic value has to be mapped into a numeric one. The numeric values (see Table 1) are not as important as their relative difference, since all the similarity scores are normalized to the interval $[0,1]$.

For stylistic distinctions, the degree of similarity is one minus the absolute value of the difference between the style values.

$$Sim_{sty}(d_1,d_2) = 1.0 - |Style(d_2) - Style(d_1)|$$

For attitudinal distinctions, similarity depends on the frequencies and the attitudes. The similarity of two frequencies is one minus their absolute differences. For the attitudes, their strength is taken into account.

$$\begin{aligned} Sim_{att}(d_1,d_2) &= S_{freq}(d_1,d_2) \cdot S_{att}(d_1,d_2) \\ S_{freq}(d_1,d_2) &= 1.0 - |Freq(d_2) - Freq(d_1)| \\ S_{att}(d_1,d_2) &= 1.0 - |Att(d_2) - Att(d_1)| \ / \ 6 \\ Att(d) &= Attitude(d) + \mathrm{sgn}(Attitude(d)) \cdot Strength(d) \end{aligned}$$

The similarity of two denotational distinctions is the product of the similarities of their three components: frequency, indirectness, and conceptual configuration. The first two scores are calculated as for the attitudinal distinctions. The computation of conceptual similarity (S_{con}) will be discussed in the next section.

$$Sim_{den}(d_1,d_2) = S_{freq}(d_1,d_2) \cdot S_{lat}(d_1,d_2) \cdot S_{con}(d_1,d_2)$$

$$\begin{aligned} S_{lat}(d_1,d_2) &= 1.0 - |Lat(d_2) - Lat(d_1)| \ / \ 8 \\ Lat(d) &= Indirectness(d) + Strength(d) \end{aligned}$$

Examples of computing the similarity between distinctions are presented in Figure 4.

6 Similarity of conceptual configurations

Peripheral concepts in Xenon are complex configurations of concepts. The conceptual similarity function S_{con} is in fact the similarity between two interlingual representations t_1 and t_2. Examples of computing the similarity of conceptual configurations are presented in Figure 5. Equation 2 computes similarity by simultaneously traversing the two representations.

$$S_{con}(t_1,t_2) = \begin{cases} S(\text{concept}(t_1),\text{concept}(t_2)) \text{ if } N_{1,2} = 0 \\ \alpha S(\text{concept}(t_1),\text{concept}(t_2)) + \\ \quad \beta \frac{1}{N_{1,2}} \sum_{s_1,s_2} S_{con}(s_1,s_2) \quad \text{otherwise} \end{cases} \qquad (2)$$

In equation 2, concept(C), where C is a interlingual representation, is the main concept (or word) in the representation. The first line corresponds to the situation when there are only main concepts, no roles. The second line deals with the case when there are roles. There could be some roles shared by both representations, and there could be roles appearing only in one of them. $N_{1,2}$ is the sum of the number of shared roles and the number of roles unique to each of the representations (at the given level in the interlingua). s_1 and s_2 are the values of any shared role. α and β are weighting factors. If $\alpha = \beta = 0.5$, the whole sub-structure is weighted equally to the main concepts.

The similarity function S deals with the case in which the main concepts are atomic (words or basic concepts) or when they are an OR or AND of complex concepts. If both are disjunctions, $C_1 =$ (OR $C_{11} \cdots C_{1n}$), and $C_2 =$ (OR $C_{21} \cdots C_{2m}$), then $S(C_1,C_2) = max_{i,j} \ S_{con}(C_{1i},C_{2j})$. The components could be atomic or they could be complex concepts; that's why the S_{con} function is called recursively. If one of them is atomic, it can be viewed as a disjunction with one element, so that the previous formula can be used. If both are conjunctions, then the formula computes the maximum of all possible sums of pairwise combinations. If $C_1 =$ (AND C_{11} $C_{12} \cdots C_{1n}$), and $C_2 =$ (AND C_{21} $C_{22} \cdots C_{2m}$), then the longest conjunction is taken. Let's say $n \geq m$ (if not the procedure is similar). All the permutations of the components of C_1 are considered, and paired with components of C_2. If some components of C_1 remain without pair, they are paired with null (and the similarity between an atom and null is zero). Then the

Frequency		Indirectness		Attitude		Strength		Style	
never	0.00	suggestion	2	pejorative	-2	low	-1	low	0.0
seldom	0.25	implication	5	neutral	0	medium	0	medium	0.5
sometimes	0.50	denotation	8	favorable	2	high	1	high	1.0
usually	0.75								
always	1.00								

Table 1: The functions that map symbolic values to numeric values.

if $d_1 = (lex_1$ low formality) and $d_2 = (lex_2$ medium formality) then $Sim(d_1,d_2) = 1 - |0.5 - 0| = 0.5$

if $d_1 = (lex_1$ always high favourable :agent) and $d_2 = (lex_2$ usually medium pejorative :agent) then
$Sim(d_1,d_2) = S_{freq}(d_1,d_2) \cdot S_{att}(d_1,d_2) = (1 - |0.75 - 1|) \cdot (1 - |(-2-0) - (2+1)|) \; / \; 6 = 0.125$

if $d_1 = (lex_1$ always medium implication $P_1)$ and $d_2 = (lex_2$ seldom medium suggestion $P_1)$ then
$Sim(d_1,d_2) = S_{freq}(d_1,d_2) \cdot S_{lat}(d_1,d_2) \cdot S_{con}(d_1,d_2) = (1 - |0.25 - 1|) \cdot (1 - (2+0+5+0) \; / \; 8) \cdot 1 = 0.03$

Figure 4: Examples of computing the similarity of lexical distinctions.

if C_1 = (C1 / departure :MOD physical and :PRE-MOD unusual) and C_2 = (C2 / departure :MOD physical)
then $S_{con}(C_1,C_2) = 0.5 \cdot 1 + 0.5 \cdot 1/2 \cdot (0.5 \cdot 1) = 0.625$

if C_1 = (C1 / person :AGENT_OF (A / drinks :MOD frequently) and C_2 = (C2 / person :AGENT_OF (A / drinks)) then $S_{con}(C_1,C_2) = 0.5 \cdot 1 + 0.5 \cdot 1/1 \cdot (0.5 + 0.5 \cdot 1/2 \cdot 1) = 0.875$

if C_1 = (C1 / occurrence :MOD (OR embarrassing awkward)) and C_2 = (C2 / occurrence :MOD awkward)
then $S_{con}(C_1,C_2) = 0.5 \cdot 1 + 0.5 \cdot 1/1 \cdot 1 = 1.0$

if C_1 = (C1 / (AND spirit purpose) :MOD hostile) and C_2 = (C2 / purpose :MOD hostile)
then $S_{con}(C_1,C_2) = 0.5 \cdot (1+0)/2 + 0.5 \cdot 1 = 0.75$

Figure 5: Examples of computing the similarity of conceptual configurations.

Experiment	No. of cases	Correct by default	Correct	Ties	Base-line %	Accuracy (no ties) %	Accuracy (total) %	Acc. nd %
Test1 Simple sentences (dev. set)	32	5	27	4	15.6	84.3	96.8	95.6
Test2 Simple sentences (test set)	43	6	35	5	13.9	81.3	93.0	84.3
Test3 French – English (test set)	14	5	7	2	35.7	50.0	64.2	28.5
Test3 English – English (test set)	14	5	14	0	35.7	100	100	100
Test4 French – English (test set)	50	37	39	0	76.0	78.0	78.0	15.3
Test4 English – English (test set)	50	37	49	0	76.0	98.0	98.0	92.3

Table 2: Xenon evaluation experiments and their results.

similarity of all pairs in a permutation is summed and divided by the number of pairs, and the maximum (from all permutations) is the resulting score: $S(C_1,C_2) = max_{p \in perms}\ \frac{1}{n}\ (\sum_{k=1}^{m}\ S_{con}(C_{1p_k},C_{2k}) + \sum_{k=m+1}^{n}\ S_{con}(C_{1k},\text{null}))$.

Here is a simple example to illustrate this procedure: $S_{con}((\text{AND}\,a\,b\,c)(\text{AND}\,b\,a)) = \frac{1}{3}\, max(S_{con}(a,b) + S_{con}(b,a) + S_{con}(c,\text{null}),\ \ S_{con}(a,a) + S_{con}(b,b) + S_{con}(c,\text{null})) = \frac{1}{3}\, max(0+0+0, 1+1+0) = 0.66$

The similarity of two words or two atomic concepts is computed from their positions in the ontology of the system. A simple approach would be this: the similarity is 1 if they are identical, 0 otherwise. But we have to factor in the similarity of two words or concepts that are not identical but closely related in the ontology. We implemented a measure of similarity for all the words, using the Sensus ontology[2]. Two concepts are similar if there is a link of length one or two between them in Sensus. The degree of similarity is discounted by the length of the link. The similarity between a word and a concept is given by the maximum of the similarities between all the concepts (senses) of the word and the given concept. The similarity of two words is given by the maximum similarity between pairs of concepts corresponding to the words. Before looking at the concepts associated with the words, stemming is used to see if the two words share the same stem, in which case the similarity is 1. This enables similarity across parts-of-speech.

7 Evaluation of Xenon

The main components of Xenon are the near-synonym choice module and HALogen. An evaluation of HALogen was already presented by (Langkilde-Geary 02a). Here, we evaluate the near-synonym choice module and its interaction with HALogen.

We conducted two kinds of evaluation experiments. The first type of experiment (Test1 and Test2) feeds Xenon with a suite of inputs: for each test case, an interlingual representation and a set of nuances. The set of nuances correspond to a given near-synonym. A graphic depiction of these two tests is shown in Figure 6. The sentence generated by Xenon is considered correct if the expected near-synonym was chosen. The sentences used in Test1 and Test2 are very simple; therefore, the interlingual representations were easily built by hand. In the interlingual representation, the near-synonym was replaced with the corresponding meta-concept.

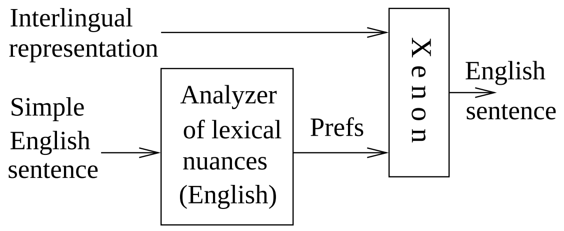
Interlingual representation
Simple English sentence
Analyzer of lexical nuances (English)
Prefs
Xenon
English sentence

Figure 6: The architecture of Test1 and Test2.

The analyzer of lexical nuances for English simply extracts the distinctions associated with a near-synonym in the LKB of NS. Ambiguities are avoided because the near-synonyms in the test sets are members in only one of the clusters used in the evaluation.

In Test1, we used 32 near-synonyms that are members of the 5 clusters presented in Figure 9. Test1 was used as a development set, to choose the exponent k for the function that translated the scale of the weights. As the value of k increased (staring at 1) the accuracy on the development set increased. The final value chosen for k was 15. In Test2, we used 43 near-synonyms selected from 6 other clusters, namely the English near-synonyms from Figure 10. Test2 was used only for testing, not for development.

The second type of experiment (Test3 and Test4) is based on machine translation. These experiments measure how successful the translation of near-synonyms from French into English and from English into English is. The machine translation experiments were done on French and English sentences that are translations of each other, extracted from the Canadian Hansard (1.3 million pairs of aligned sentences from the official records of the 36th Canadian Parliament)[3]. Xenon should generate an English sentence that contains an English near-synonym that best matches the nuances of the initial French near-synonym. If Xenon chooses exactly the English near-synonym used in the parallel text, this means that Xenon's behaviour was correct. This is a conservative evaluation measure, because there are cases in which more than one translation is correct.

The French–English translation experiments take French sentences (that contain near-synonyms of interest) and their equivalent English translations. We can assume that the interlingual representation is the same for the two sentences. Therefore, we can use the interlingual representation for the English sentence

[2]We could have used an off-the-shelf semantic similarity package, such as the one provided by Ted Pedersen (http://www.d.umn.edu/∼tpederse/tools.html) or the one described in (Budanitsky & Hirst 01), but it contains similarity measures mainly for nouns (on the basis of WordNet's noun hierarchy), and it would be time-consuming to call it from Xenon.

[3]http://www.isi.edu/natural-language/download/hansard/

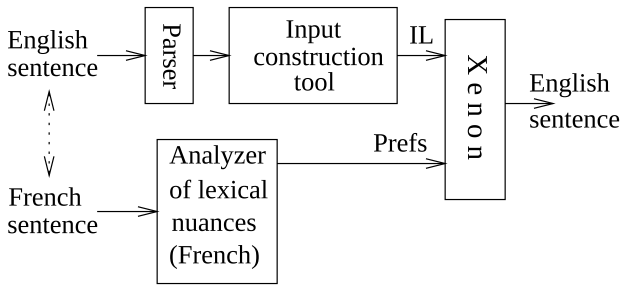

Figure 7: The architecture of Test3 and Test4, the French-to-English part.

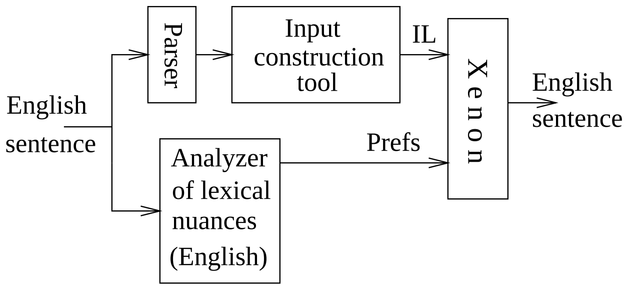

Figure 8: The architecture of Test3 and Test4, the English-to-English part.

to approximate the interlingual representation for the French sentence. As depicted in Figure 7, the interlingual representation is produced using an existing tool that was previously used by (Langkilde-Geary 02a) in HALogen's evaluation experiments. In order to use this input construction tool, we parsed the English sentences with Charniak's parser (Charniak 00), and we annotated the subjects in the parse trees. In the interlingual representation produced by the input construction tool we replaced the near-synonym with the corresponding meta-concept. For Test4, the baseline is much higher because of the way the test data was collected. The analyzer of French nuances from Figure 7 needs to extract nuances from a LKB of French synonyms. We created by hand a LKB for six clusters of French near-synonyms (those from Figure 10), from two paper dictionaries of French synonyms, Bénac (Bénac 56) and Bailly (Bailly 73).

A similar experiment, translating not from French into English but from English into English via the interlingual representation, is useful for evaluation purposes. An English sentence containing a near-synonym is processed to obtain its semantic representation (where the near-synonym is replaced with a meta-concept), and the lexical nuances of the near-synonym are fed as preferences to Xenon. Ideally, the same near-synonym as in the original sentence would be chosen by Xenon. The percentage of times this happens is an evaluation measure. The architecture of these experiments is presented in Figure 8.

Test4 is similar to Test3, but instead of having one sentence for a near-synonym, it contains all the sentences in a given fragment of Hansard (4 Megabytes of House debates) in which words of interest occurred. Therefore, Test4 has the advantage of containing naturally occurring text, not artificially selected sentences. It has the disadvantage of lacking those near-synonyms in the test set that are less frequent. The English and French near-synonyms used in Test3 and Test4 are the ones presented in Table 10.

In order to measure the baseline (the performance that can be achieved without using the LKB of NS), we ran Xenon on all the test cases, but this time with no input preferences. Some of the choices could be correct solely because the expected near-synonym happens to be the default one.

The results of the evaluation experiments are presented in Table 2. For each test, the second column shows the number of test cases. The column named "Correct" shows the number of answers considered correct. The column named "Ties" shows the number of cases of ties, that is, cases when the expected near-synonym had weight 1.0, but there were other near-synonyms that also had weight 1.0, because they happen to have the same nuances in the LKB of NS. In such cases, Xenon cannot be expected to make the correct choice. More precisely, the other choices are equally correct, at least as far as Xenon's LKB is concerned. Therefore, the accuracies computed without considering these cases (the seventh column) are underestimates of the real accuracy of Xenon. The eighth column presents accuracies while taking the ties into account, defined as the number of correct answers divided by the difference between the number of cases and the number of ties.

There are two reasons to expect Xenon's accuracy to be less then 100%. The first is that there are cases in which two or more near-synonyms get an equal, maximal score because they do not have any nuances that differentiate them (either they are perfectly interchangeable, or the LKB of NS does not contain enough knowledge) and the one chosen is not the expected one. The second reason is that sometimes Xenon does not choose the expected near-synonym even if it is the only one with maximal weight. This may happen because HALogen makes the final choice by combining the weight received from the near-synonym choice module with the probabilities from its language model. Frequent words may have high

English (n.): benefit, advantage, favor, gain, profit
English (v.): flow, gush, pour, run, spout, spurt, squirt, stream
English (adj.): deficient, inadequate, poor, unsatisfactory
English (adj.): afraid, aghast, alarmed, anxious, apprehensive, fearful, frightened, scared, terror-stricken
English (n.): disapproval, animadversion, aspersion, blame, criticism, reprehension

Figure 9: Near-synonyms used in the evaluation of Xenon (Test1).

English (n.): mistake, blooper, blunder, boner, contretemps, error, faux pas, goof, slip, solecism
French: erreur, égarement, illusion, aberration, malentendu, mécompte, bévue, bêtise, blague, gaffe, boulette, brioche, maldonne, sophisme, lapsus, méprise, bourde

English (n.): alcoholic, boozer, drunk, drunkard, lush, sot
French: ivrogne, alcoolique, intempérant, dipsomane, poivrot, pochard, sac à vin, soûlard, soûlographe, éthylique, boitout, imbriaque

English (v.): leave, abandon, desert, forsake
French: abandonner, délaisser, déserter, lâcher, laisser tomber, planter là, plaquer, livrer, céder

English (n.): opponent, adversary, antagonist, competitor, enemy, foe, rival
French: ennemi, adversaire, antagoniste, opposant, détracteur

English (adj.): thin, lean, scrawny, skinny, slender, slim, spare, svelte, willowy, wiry
French: mince, élancé, svelte, flandrin, grêle, fluet, effilé, fuselé, pincé

English (n.): lie, falsehood, fib, prevarication, rationalization, untruth
French: mensonge, menterie, contrevérité, hâblerie, vanterie, fanfaronnade, craque, bourrage de crâne

Figure 10: Near-synonyms used in the evaluation of Xenon (Test2,3,4).

probabilities in the language model. If the expected near-synonym is very rare, or maybe was not seen at all by the language model, its probability is very low. When combining the weights with the probabilities, a frequent near-synonym may win even if it has a lower weight assigned by the near-synonym choice module. In such cases, the default near-synonym (the most frequent of the cluster) wins. Sometimes such a behaviour is justified, because there may be other constraints that influence HALogen's choice.

Table 2 also includes the results for the baseline experiments. For Test1 and Test2 the baseline is much lower than Xenon's performance. For example, for Test1, Xenon achieves 96.8% accuracy, while the baseline is 15.6%. An exception is the baseline for Test4, which is high (76%), due to the way the test data was collected: it contains sentences with frequent near-synonyms, which happen to be the ones Xenon chooses by default in the absence of input preferences. For Test3 and Test4 the baseline is the same for the French and English experiments because no nuances were used as input.

Another way to measure the performance of Xenon is to measure how many times it makes appropriate choices that cannot be made by HALogen, that is, cases that make good use of the nuances from the LKB of NS. This excludes the test cases with default near-synonyms, in other words the cases when Xenon makes the right choice due to the language model. It also excludes the cases of ties when Xenon cannot make the expected choice. The last column in Table 2 shows accuracies for these "non-default" cases. For the experiments with only English near-synonyms, Xenon is performing very well, managing to make correct choices that cannot be made by default. Accuracies vary from 92.3% to 100%. For the experiments involving both French and English experiments, Xenon makes only a few correct choices that cannot be made by default. This is due to the fact that there is some overlap in nuances between French and English synonyms, but most of the overlap happens for the near-synonyms that are defaults.

8 Conclusion

Xenon can successfully choose the near-synonym that best matches a set of input preferences. In future work we plan to extend the near-synonym choice to take into account the collocational behaviour of the near-synonyms.

Acknowledgments

We thank Irene Langkilde-Geary for useful discussions and for making the input construction tool available. We thank Phil Edmonds for making available the source code of I-Saurus. Our work is financially supported by the Natural Sciences and Engineering Research Council of Canada and the University of Toronto.

References

(Bailly 73) René Bailly, editor. *Dictionnaire des Synonymes de la Langue Française*. Larousse, Paris, 1973.

(Bénac 56) Henri Bénac, editor. *Dictionnaire des Synonymes*. Librarie Hachette, Paris, 1956.

(Budanitsky & Hirst 01) Alexander Budanitsky and Graeme Hirst. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Proceedings of the Workshop on WordNet and Other Lexical Resources, Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2001)*, Pittsburgh, USA, 2001.

(Charniak 00) Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics and the 6th Conference on Applied Natural Language Processing (NAACL-ANLP 2000)*, Seattle, USA, 2000.

(Edmonds & Hirst 02) Philip Edmonds and Graeme Hirst. Near-synonymy and lexical choice. *Computational Linguistics*, 28 (2):105–145, 2002.

(Edmonds 99) Philip Edmonds. *Semantic representations of near-synonyms for automatic lexical choice*. Unpublished PhD thesis, University of Toronto, 1999.

(Hayakawa 94) S. I. Hayakawa, editor. *Choose the Right Word*. Second Edition, revised by Eugene Ehrlich. HarperCollins Publishers, 1994.

(Inkpen & Hirst 01) Diana Zaiu Inkpen and Graeme Hirst. Building a lexical knowledge-base of near-synonym differences. In *Proceedings of the Workshop on WordNet and Other Lexical Resources, Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2001)*, pages 47–52, Pittsburgh, USA, 2001.

(Inkpen 03) Diana Inkpen. *Building a Lexical Knowledge-Base of Near-Synonym Differences*. Unpublished PhD thesis, (In preparation), University of Toronto, 2003.

(Knight & Luk 94) Kevin Knight and Steve Luk. Building a large knowledge base for machine translation. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, Seattle, WA, 1994.

(Langkilde & Knight 98) Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics joint with 17th International Conference on Computational Linguistics (ACL-COLING'98)*, pages 704–712, Montreal, Quebec, Canada, 1998.

(Langkilde-Geary 02a) Irene Langkilde-Geary. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the 12th International Natural Language Generation Workshop*, pages 17–24, New York, USA, 2002.

(Langkilde-Geary 02b) Irene Langkilde-Geary. *A Foundation for a General-Purpose Natural Language Generation: Sentence Realization Using Probabilistic Models of Language*. Unpublished PhD thesis, University of Southern California, 2002.