BADGER: A New Machine Translation Metric

Steven Parker
BabbleQuest.org
East Lansing, Michigan
sbparke@babblequest.org

Abstract

With the introduction of new and more sophisticated MT systems, increased interest has been focused on their potential applications and relative performance. The number of viable commercial and academic systems to choose from has created a dilemma for commercial and government decision makers. Objective automated measurement of MT system performance is becoming increasingly desirable. In response, NIST has sponsored the first MT metric challenge in 2008 to help focus development in this area. Here we present new research in response to this challenge. This paper introduces BADGER, a new language independent translation metric based on compression and information theory. While the test data provided is too small for a definitive performance analysis, we show this metric's correlation with human judgment at the segment level. We show this approach is significantly different from one n-gram approach as well as showing its similarity with word error rate measures. We also present a number of input normalization techniques centered on an efficient language independent Holographic Reduced Representation and report on the relative performance of several word normalization methods within this data set.

1 Introduction

We introduce a new alignment free method based on information theory and data compression. Our method differs significantly from many n-gram approaches currently in use. We implement an efficient Normalized Compression Distance (NCD) utilizing the Burrows Wheeler Transformation (BWT). The BWT enables us to take into account common sentence contexts with no limit on the size of these contexts. We explore this distance measure and its relationship with n-gram and edit distance measures already in use.

We also report on the effect of a variety of language independent word normalization methods. The majority of these normalization methods utilize a simple Holographic Reduced Representation (HRR) we call a SpatterMap. Relationships between words within the target language are represented in a compact binary vector whose relative distance can be easily computed by cosine similarity (Kanerva, 1996). These semantic similarities are used as an ordering function within the BWT as well as a filter for several normalization methods including stemming, spelling variations and semantic relationships.

2 Description of the Metric

The BADGER Metric is based on a simple compression distance calculation. Word normalization is performed using a variety of methods that rely on a Holographic Representation to support language independence. Here we will present a high level description of BADGER.

2.1 HRR Word Model

Metrics such as METEOR and many of the proposed extensions to existing metrics suggest the use or future use of a lexicon such as WordNet. Unfortunately these resources only exist for a small number of languages and in addition many do not provide the coverage of WordNet. We strive to extend our metric with a pipeline of matching modules similar to those employed by METEOR but using language independent methods. We have

used unsupervised methods to achieve our WordNet analog.

We have been investigating the family of HRRs to achieve improved performance on other NLP tasks such as topic identification and query expansion. The use of these representations in a MT metric extends the set of techniques we can deploy in this area, and also provides an excellent laboratory to test the potential efficacy of this technique in other areas of text processing.

The HRR type chosen was a simple ternary version called a SpatterVector. We have created a map structure containing a set of SpatterVectors to generate a new data structure called a SpatterMap. Each word in the corpus is assigned a vector containing a sparse set of ternary values (-1,0,1). Each vector contains 4 ternary values.[1] Two values are ones and two are negative ones. These vectors interact with each other using the operation below.

A	B	Interaction Operation (Θ)
0	1	1
0	-1	-1
1	-1	0
1	1	1
-1	-1	-1

Table 1. Ternary Logic as used by the HRR

The SpatterMap is simply the collection of these sparse vectors and their interactions for each word in the corpora given some context as described in (Kanerva et al, 2000). A fixed context length within a single sentence was chosen for each word.

Word Index length	14,000
Ternary values	4
Context length	4 preceding 4 following

Table 2. SpatterMap parameters used

[1] Ternary values are called trits

Input:
M =Vector of words in corpora
U = Unique words in corpora
N = Length of SpatterVector
S = SpatterMap[U][N]
C = Context Length

Initalize: For i = 0 to U
Create sparse ternary vector W_i[0..N]
Place 2 each of (1,-1) at random within W_i
Create unique id for word $W_{(j=i)}$

Compute: For each occurrence of word W_j in corpora M
If W_i appears within C of W_j
$S[j] = W_i \Theta S[j]$

Ouput: SpatterMap S

Figure 1. Algorithm for SpatterMap construction

Once the vector map is constructed, similarities between words can be calculated. We use the normalized cosine distance described in (Schone, Jurafsky, 2000).

The SpatterMap data consists of words (including stop words) contained within the text corpora. Additionally log likelihood (Dunning, 1993) was used to extract potential collocations from the data. These are also included in the SpatterMap. Rare words were not included in the SpatterMap.

2.2 Word Normalization

The SpatterMap above was utilized to perform a number of word normalizations. Stemming was accomplished by using the Levenshtein distances between words in the reference and system sentences. If the threshold was within some percentage of the total length of the word we then used the normalized cosine distance between words to decide if this word was likely an **inflection**. Rare words not in the SpatterMap were assumed to be either **entities** or **transliterated** words. If the Levenshtein distance of these words were within a threshold percentage of the word length these words were considered to be also equivalent. **Collocations** and **single words** not otherwise matched were measured by normalized cosine similarity and if they were within a restricted percentile these were also considered equivalent. This provided our measure of semantic similarity. The mapping of collocations and single words is unique to BADGER. No effort was made to perform soft matching or provide weightings proportional to this

cosine distance. Over matching of potentially normalized words is avoided by allowing only one match per unique word.

Collocations	Semantic Relations
monday_night	the=an
armed_men	an=the
police_forces	of=in
early_hours	**train=times**
deputy_minister	baghdad=capital
drug_trafficking	was=of
drinking_water	**monday=tuesday**
economic_growth_rate	early_hours=night
education_minister	officer=rank
colin_powell	colonel=officer
clint_eastwood	**baghdad=security**
	security=today
	in=south
	to=south

Table 3. Normalizations from one sample sentence.

Inflections	Entities
consists=consisted	forik=furik
brigade=brigades	itamar=itmar
border=borders	fourik=furik
bird=birds	braverman=brafirman
explosive=explosives	haneya=haniya
want=wants	hezbollah=hizballah
believes=believed	kesra=kasra
	khayyun=khayoun
	kosatcheov=kosachov

Table 4. Normalizations from selected sentences.

While the modules for inflection and entity normalization seem to work well, the collocation and semantic mapping modules may need further testing and refinement.

2.3 Burrows Wheeler Transform

While the above refinements could have easily been applied to the BLEU or METEOR metrics, a large body of work on n-gram based metrics already exists. In the spirit of trying something unique we wished to explore a non n-gram approach. Our vision for a metric would ideally be a block edit distance or Kolmogorov distance measure. As these approaches are NP complete or incomputable respectively, we turned to algorithms utilized for DNA sequence comparison. After looking at several context free methods we settled on the Burrows Wheeler Transformation (BWT) as much for its matching potential as its ease of implementation and overall speed.

This algorithm lies at the heart of the popular bzip2 algorithm and has recently been applied to DNA sequence analysis. The essence of the algorithm is that an input string is taken and rotated counterclockwise by one element to generate a set of rows. This set of rows is then lexically sorted yielding rows grouped by matching context. Our metric merges the reference and system translations into a single list that is used to compute the potential compression. This result is compared with the resulting compression of the reference and translation lists to generate a relative distance metric. The elements on the trailing edge that do not equal the preceding element generate a sum that yields the BWT's potential compression.

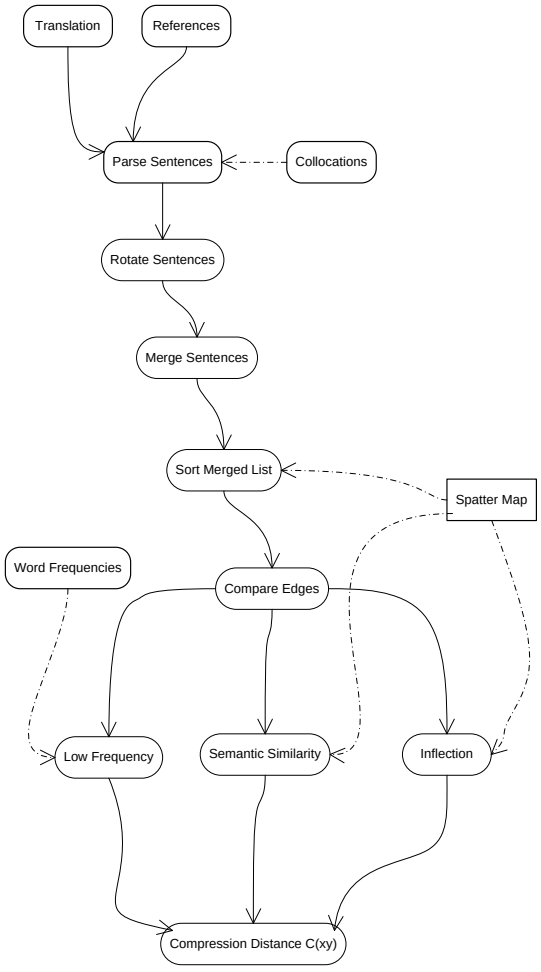

Figure 2. Data Flow for concatenated references and translation

Below is an example of the BWT on the words area and read.

Input	Rotations	Sorted Rows	Edge	C(xy) = 5
AREA READ	AREA	AARE	**E**	1
	REAA	ADRE	**E**	0
	EAAR	AREA	***A***	1
	AARE	DREA	***A***	0
	READ	EAAR	***R***	1
	EADR	EADR	***R***	0
	ADRE	REAA	A	1
	DREA	READ	D	1

Table 5. BWT example

We use the transformation and calculate the potential compression possible for a set of strings. One of the main advantages for using this transform is that we can utilize contexts over the entire set of reference sentences.

To create a metric we use the following compression distance described in (Án et al) This yields a value in the range [0,1].[2]

$$NCD(x,y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}$$

C(x) Compression for x.
C(xy) Compression for concatenation of x and y.

We set y to be the reference sentence or set of sentences. This allows us to compare more then one reference to the MT output at one time.

The equation above is an approximation of the Normalized Information Distance based on Kolmogorov complexity.

$$NID(x,y) = \frac{K(x,y) - \min\{K(x), K(y)\}}{\max\{K(x), K(y)\}}$$

Where K(x) is the shortest program that outputs the string x and halts on a universal Turing machine.

It is worth mentioning that this approximation increases in accuracy with the amount of data provided. The initial application of this approach for DNA sequence alignment uses sequence lengths typically from 5,000 to 3 billion nucleotides[3]. Given the segment lengths we are measuring we are faced with a data sparcity problem. We attempt to alleviate this issue with some extensions to the BWT transformation.

2.4 Extensions to BWT

While the lexical ordering of the rotated sequences yields a set of matching contexts, it becomes apparent that other ordering may be useful to extract other similarities. This may also help to alleviate the data sparicity issue. You can see the potential of this alternative ordering approach by examining table 3. We investigate an alternative ordering using semantic distance in addition to lexical ordering to attempt to extract additional information from the transformation. The HRR above is again called into service to yield an ordered set of words from both the reference and target sentences utilizing a simple hierarchal clustering. The resulting tree is traversed to yield an ordered set that is then used to sort the rotated sequences. Below are sample orderings from the data set.

```
a, added, announcements, are, as,
await, be, by, command, Egyptian, fa-
tah, for, formal, from, general, he,
in, institutions, issued, leadership,
leading, middle_east, movement, news,
news_agency, of, official,
press_agency, quoted, reply, reported,
response, s, saying, statements, that,
the, to, wait, waiting, was, we, will
```

Figure 3. Lexical word ordering

```
added, press_agency, s, await, by, re-
ply, middle_east, we, are, from, as,
leading, a, news_agency, issued, an-
nouncements, news, general, for, will,
be, was, command, to, he, that, in,
movement, the, of, Egyptian, institu-
tions, reported, waiting, wait, re-
sponse, fatah, leadership, official,
formal, statements, quoted, saying
```

Figure 4. Semantic word ordering

[2] We invert this value to allow 1 to be the highest score which we believe is more intuitive. The range is between 0 and 1 in most cases.

[3] 3 billion nucleotides for the human genome

To provide a smaller memory footprint we also created a lite version of BADGER which does not include the HRR model. In this case the alternative ordering ranks sentences by the highest number of matching words. We call this new sort order maximal match. The performance of this new ordering surprisingly outperformed the semantic orderings in some cases. Unfortunately, this ordering was discovered within the last weeks of the submission period as we removed the HRR model. In depth analysis of this ordering has not been done.

Matching on the edge of the sequences is accomplished by applying the word normalization modules already discussed. In the future we intend to utilize additional relationships such as syntactic relations.

After a small amount of ad hoc testing against the development set we have settled on the relative weights below. The simple matching measure is simply the percentage of unigrams matched across all references. This is used because the BWT is not bijective and can result in a score of zero for some translation samples even if some unigrams are in common. BWT measurement alone may not discriminate between systems with poor output quality.

Ordering	Weight
Lexical Ordering	17%
Semantic Ordering	80% (full version)
Maximal match	80% (lite version)
Simple Matching	3%

Table 6. BWT ordering weights

These additional orderings help somewhat to deal with the relatively short data sets we are comparing. Other approaches may involve alternative ordering other than simple rotation. It has yet to be seen if there are actual orderings that would work in practice.

3 Implementation

The BADGER metric is written in Java and can run on many common operating systems including Linux, Solaris, Windows, and MacOS. The metric software and supporting files can easily be packaged into a single executable jar file for ease of installation. The system can parse and auto correlate a set of XML files following the NIST DTD. The system itself also provides a command line and graphical interface.

The most computationally expensive part of the metric is the calculation of the normalized cosine distance. Eight hundred random samples are used to calculate the appropriate percentile cutoff for word similarity. Extensive caching and threading are used to efficiently calculate these samples which can easily be performed in parallel. Significant speed improvements can be gained by simply precomputing the sample distances. This precomputation step would take a few days to perform. To provide a smaller memory foot print we have also created the BADGER Lite version which does not utilize the HRR representation.

A number of open source libraries were used to implement this metric including Colt, Apache Commons (Math, Collections and CLI), and EhCache. The Wikipedia and web pages were parsed with the JAMWiki and Jericho.

The implementation is available now as a free utility. The source code will be released as a separate package in the near future.

3.1 Model Training Set

Training of the word model was performed on a corpora mined from the internet. Data from the wikipedia.org and a set of about 250K news and blog web pages were added to round out the text collection. The total parsed and normalized data set was about 143 million words. The resulting input data was far from perfectly clean but was essentially text with XML and HTML markup removed after processing was completed.

The SpatterMap can either be contained entirely in memory during training as in the models delivered for evaluation or can be cached onto the systems hard drive yielding very large memory models. The initial models were approximately 30 Gig in size. The final models after reducing the overall spatter vector lengths, overall sentence contexts and lower frequency words was about 300 Meg. Training time was about 4 days for the very large models and about 12 hours for the smaller models. Compared to SVD techniques this approach is very attractive. The very large models utilized the open source EHCache package for caching data to disk.

Collocation candidates were extracted using log likelihood. False positives were controlled

by removing words by frequency and enforcing a distribution measure that ensured local references did not get undue credit.

4 Metric Behavior

In the future the combination of several metrics into an ensemble of metrics would be desirable as each method should have different failure modes. Metrics with differing behaviors could be combined through some form of machine learning method. Knowing which "family" each metric belongs to will ease the selection of representative metrics for inclusion in this heterogeneous approach. Our intent in this section is to study the behavior of BADGER in relation to the METEOR and word error rate (WER) metrics. [4]

To compare the behaviors of the metrics above we flattened the development data set into one large document set with monotonically increasing segment ids. This allowed for simple comparisons between the metric results. As the results below show the overall correlation of BADGER and METEOR is low and the correlation of BADGER and WER is high. [5]This stands to reason as BADGER is an approximation of a block edit distance and should correlate well with the WER which is based on simple edit distance. As the dataset provided for testing is Arabic to English and the fact that Arabic has similar word order to in comparison with other languages (e.g. Chinese and Urdu), we can expect this correlation to be even highier. We are very interested to see what this correlation would be when comparing these metrics on a more varied set of language pairs.

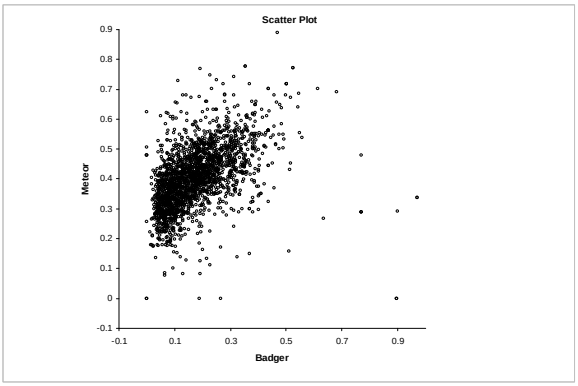

Figure 6. Pearson Correlation 0.47
METEOR Vs BADGER

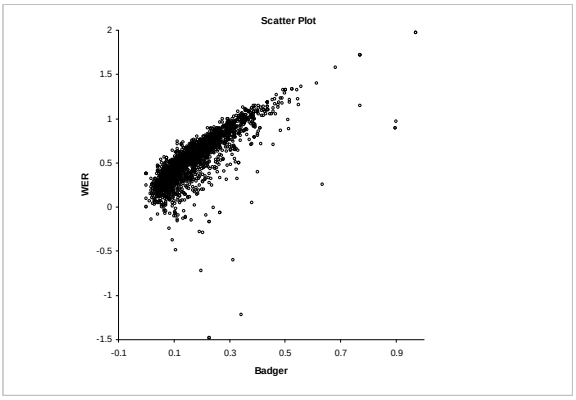

Figure 7. Pearson Correlation 0.79
WER Vs BADGER

As one can see the BADGER metric is significantly different from an n-gram approach shown in figure 1. And BADGER exhibits behavior more in line with word error rates as shown in figure 2.

5 Initial Results

As stated above the relatively small amount of data provided in the development set does not lend itself to making any definitive conclusions as the confidence interval tends to overlap the differences between metrics. BADGER does seem to be slightly behind the other metrics selected for measurement.

Metric Name	Preferences	Adequacy
Meteor	.56110	.59780
WER	.61669	.62374
BADGER	.54455	.51358
BADGER Lite	.53448	.56198

Table 7. Initial Results

[4] We were not able to analyze the differences between BADGER and the BLEU or NIST products as the correlation between these systems was not trivial. (due to segment ids not being preserved).

[5] Pearson correlation between Meteor and WER is .52

6 Impact of Normalization

Extensive testing was done in order to determine the relative value of the normalizations performed within BADGER. We present some of the results below. The entire test set of 130 tests was too large to include in this report. The row "All" below is the version delivered to NIST for evaluation. We did roll the dice a bit in selecting this set of normalizations as the table below suggests that normalization of collocations and semantic relations may actually be detrimental to the performance of our metric. However, after looking at the measures for METEOR we could not remove them with confidence from the final delivery. We are assuming for lower density languages these features will have a more positive impact.

Normalization	Preference	Adequacy
All	.54455	.51358
No Collocations	.55251	.53783
No Entity/Translit	.53973	.51570
No Semantic Match	.55319	.55827
No Inflection	.54601	.51350
No Collocations or Semantic Match	.58401	.56865
Nothing	.58259	.57214

Meteor exact	.60441	.70615
Meteor wn_stem	.59726	.71706
Meteor wn_syn	.56110	.59780

Table 8. Normalizations

7 Impact of BWT orderings

In order to see the effects of each ordering by itself we ran additional tests to see what effect these might have. Note in the table below no normalizations were performed.

Ordering	Preference	Adequacy
Lexical only	.57709	.56340
Semantic only	.57681	.56337
Word Match	.56435	.57212
Weighted	.58259	.57214

Table 7. BWT ordering effects

It is worth noting that the difference in correlation between the alternative orderings is statistically insignificant and may indicate that any further modifications to sort order may not result in an overall improvement in BADGER's performance.

8 Impact by Reference Number

Having multiple references does provide better performance for BADGER. The relative performance measures between single references and multiple references are provided below. Both versions used the normalizations and weights delivered to NIST.

Version	Preferences single	Adequacy single
Full	.49003	.49083
Lite	.53448	.52945
Version	Preferences Multiple (4)	Adequacy Multiple (4)
Full	.54455	.51358
Lite	.53929	.56198

Table 9. Impact by number of references

9 Is it Gameable?

We believe that BADGER is resistant to gaming. BADGER uses a large amount of potential context during the comparison of MT output as well as multiple scoring functions internally. In addition, the n-gram optimization used in most SMT systems would be unable to accommodate the length of context currently available to BADGER. However, tuning using the metric itself or something in the same metric family such as word error rates may result in a MT engine that is tuned to perform well for this metric. The best approach to avoidance of metric gaming will probably be the adoption of an ensemble of metric systems that utilize measurements from a variety of metric approaches.

In the future BADGER may investigate the inclusion of non-linear combinations of internal compression distances from a variety of sorting methods as well as alternative string matching algorithms. This would represent a significant increase in number of parameters to optimize. The motivation for using a non linear combination of measures is more motivated by our need to more closely correlate with human judgment and provide effective differentiation between MT systems then the avoidance of gaming specifically.

10 Future Work

BADGER is in its infancy and we expect to generate a number of enhancements. We intend to enhance the HRR representation to include neurologically motivated behaviors such as reinforcement and attention[6]. Currently our extended BWT is not bijective. We intend to enhance the word matching algorithm to enforce bijective behavior. [7]Creating bijective relations with an unconstrained set of features is problematic but we intend to reduce the word search space prior to sorting and use the bijective ordering method described in (Mantaci, 2008). We also would like to investigate some of the concepts for unsupervised grammar induction such as in (Olney, 2007). It has also been noted that segmentation of the sentence into fragments may be used to provide higher accuracy. It seems obvious that at least splitting sentences at quotation markings in our test set would have yielded better accuracy. However, this is difficult to do in a language independent way. Other reordering may be attempted that are not simple rotations but rotations with some grammatical underpinnings per language. We intend to investigate some primitive unsupervised grammar induction techniques to see if this is possible.

We will also be looking for partners to create a larger freely available test set that is either machine or human generated. We intend to investigate the generation of synthetic data to allow the control and description of the importance of inflection, argument structure, referents, stop word usage, word selection and negations. This should allow us to generate an easily readable report for the end user on the shortcomings of specific MT engines that are measured.

11 Initial NIST Results

The expected performance of the BADGER system was a Pearson correlation between .4 and .5 at the segment level. The actual performance on the primary evaluation set was below this approximately between .2 and .4. This performance is significantly below many of the submitted metrics including Meteor. This would suggest that either performance on Chinese and Persian is significantly different or a number of word normalization modules should have been left out of the final system. Performance on the secondary evaluation set was better approximately from .4 to .5 at the segment level. This tends to suggest that the some of the word normalization modules should have been left out namely collocation and matching by semantic similarity. In the secondary evaluation set the target language was French which would negate many of the normalization modules that would have caused issues. The module responsible for matching out of vocabulary words would have decreased in precision as this module would have matched most words within some edit distance. This may explain the reduction in accuracy in the secondary set for multiple references. We will require more data to establish the actual performance parameters and potential optimizations.

12 Conclusions

We have presented the BADGER MT metric and preliminary results. Although we can not make any concrete judgments as to its efficacy we are looking forward to more test results and the opportunity to extend this approach. This software is freely available for download at http://babblequest.org/badger. Source code will be released via the web once it has undergone unit testing and code review.

Acknowledgments

Many thanks to NIST for sponsoring this challenge. We would also like to thank Steven Bradtke and Carl Rubino for their valuable feedback.

References

Án,M.C.;Alfonseca, M.&Ortega,A.,COMMON PITFALLS USING THE NORMALIZED COMPRESSION DISTANCE: WHAT TO WATCH OUT FOR IN A COMPRESSOR

Banerjee, S.& Lavie, A., 2005 METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments

[6] Using random sub sampling for word vectors and variable vector lengths inversely proportional to word frequency.

[7] Create a one to one matching. Currently our normalization module may match more then one equivalent word choice from one sentence.

In Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43th Annual Meeting of the Association of Computational Linguistics (ACL-2005), Ann Arbor, 65-72

Berstel,J.,2007, Sturmian Words, Sturmian Trees and Sturmian Graphs, A Survey of Some Recent Results *CAI 2007, ThessalonikiJ*

Crochemore, M.; Desarmenien, J. & Perrin, D., 2005 A note on the Burrows-Wheeler transformation *CCSd/HAL (France), HAL - CCSd – CNRS*

Dunning, T. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics, 19(1):61—74*

Kanerva, P. Binary spatter-coding of ordered K-tuples. C. von der Malsburg, W. von Seelen, J.C. Vorbrüggen, and B. Sendhoff (eds.), *Artificial Neural NetworksICANN 96 (Proceedings 1996 International Conference,Bochum, Germany; Lecture Notes in Computer Science 1112), pp. 869-873*

Kanerva, P., Kristoferson, J., and Holst, A. "Random indexing of text samples for Latent Semantic Analysis.", *L.R. Gleitman and A.K. Josh (eds.), Proc. 22nd Annual Conference of the Cognitive Science Society (U Pennsylvania*), p. 1036.

Lavie, A.& Agarwal,A., Meteor: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments

Mantaci, S.; Restivo, A. & Sciortino, M.,2008, Distance measures for biological sequences: Some recent approaches, *Int. J. Approx. Reasoning, , 47*, 109-124

Mantaci, S.; Restivo, A.;Rosone,G.&Sciortino, M.,2008, A New Combinatorial Approach to Sequence Comparison, *Theor. Comp. Sys., Springer-Verlag New York, Inc. , 42*, 411-429

Olney, A. M.,2007, Latent Semantic Grammar Induction: Context, Projectivity, and Prior Distributions, *TextGraphs-2: Graph-Based Algorithms for Natural Language, Processing Proceedings of the Workshop*, 45-52

Schone,P. & Jurafsky, D.,2000, Knowledge-free induction of morphology using Latent Semantic Analysis, *Proc. of the Computational Natural Language Learning Conference, Lisbon,* , 67-72

Wikipedia,2008, Word error rate --- Wikipedia, The Free Encyclopedia