INFORMATION-THEORETIC METHODS FOR ANALYSIS AND INFERENCE IN ETYMOLOGY

Hannes Wettig[1], *Javad Nouri*[1], *Kirill Reshetnikov*[2] *and Roman Yangarber*[1]

[1]Department of Computer Science, University of Helsinki, Finland, First.Last@cs.helsinki.fi
[2]Academy of Sciences, Institute of Linguistics, Moscow, Russia.

ABSTRACT

We introduce a family of minimum description length models which explicitly utilizes phonetic features and captures long-range contextual rules that condition recurrent correspondences of sounds within a language family. We also provide an algorithm to learn a model from this family given a corpus of cognates, sets of genetically related words. Finally, we present an *imputation* procedure which allows us compare the quality of alignment models, as well as the goodness of the data sets. Our evaluations demonstrate that the new model yields improvements in performance, as compared to those previously reported in the literature.

1. INTRODUCTION

This paper introduces a family of context-aware models for alignment and analysis of etymological data on the level of phonetic features. We focus on discovering the rules of regular (or recurrent) phonetic correspondence across languages and determining genetic relations among a group of languages, based on linguistic data. In this work, we use the StarLing database of Uralic, [1], based on [2], restricted to the Finno-Ugric sub-family, consisting of 1898 *cognate sets*, as well as *Suomen Sanojen Alkuperä* (SSA), "The Origin of Finnish Words," a Finnish etymological dictionary, [3], which contains over 5000 cognate sets. Elements within a given cognate set are words posited by the database creators to be derived from a common origin, a word-form in the ancestral *proto-language*.

One traditional arrangement of the Uralic languages—adapted from Encyclopedia Britannica—is shown in Figure 1; alternative arrangements found in the literature include moving Mari into a separate branch, or grouping it with Mordva into a branch, called "Volgaic".

We aim to find the best *alignment* at the level of single sounds. The database itself only contains unaligned sets of corresponding words, with no notion of which sounds correspond, i.e., how the sounds align. We learn rules of phonetic correspondence allowing only the data to determine what rules underly it, using no externally supplied (and possibly biased) prior assumptions or "universal" principles—e.g., no preference to align vowel with vowels, a symbol with itself, etc. Therefore, all rules we find are *inherently encoded* in the corpus itself.

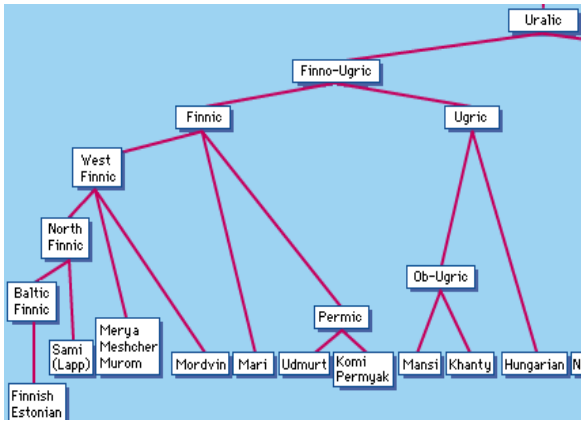

Figure 1. Finno-Ugric branch of Uralic language family

The criterion we use to choose a model (class) from the family we define is the code-length needed to communicate the complete (aligned) data. The learned minimum description length (MDL) models provide the desired alignments on the sound level, but also the underlying rules of correspondence, which enable us to *compress* the data. Apart from looking at the code-length, we also evaluate our models using an imputation (reconstruction of held-out data) procedure and by building phylogenies (family trees). We release the suite of etymological software for public use.

Most closely related to this work is our own previous work, e.g., [4], and work conducted at Berkeley, e.g., [5, 6]. The main improvement over these lies in awareness of a broader phonetic context of our models. We build decision trees to capture this context, where irrelevant context does not increase model complexity.

2. ALIGNING PAIRS OF WORDS

We begin with pairwise alignment: aligning pairs of words, from two related languages in our corpus of cognates. For each word pair, the task of alignment means finding exactly which symbols correspond. The simplest form of such alignment at the symbol level is a pair $(\sigma : \tau) \in \Sigma \times T$, a single symbol σ from the source alphabet Σ with a symbol τ from the target alphabet T. We denote the sizes of the alphabets by $|\Sigma|$ and $|T|$.

To model *insertions* and *deletions*, we augment both

alphabets with a special empty symbol—denoted by a dot—and write the augmented alphabets as $\Sigma_{\cdot}$ and $T_{\cdot}$. We can then align word pairs such as *vuosi—al* (meaning "year" in Finnish and Xanty), for example as any of:

v	u	o	s	i		v	u	o	s	i
|	|	|	|	|		|	|	|	|	|
a	l	.	.	.		.	a	.	l	.

etc...

The alignment on the right then consists of the symbol pairs: (v:.), (u:a), (o:.), (s:l), (i:.).

3. FEATURE-WISE CONTEXT MODELS

Rather than encoding symbols (sounds) as atomic, we code them in terms of their phonetic features. To this end, the corpus has been transcribed into feature vectors, where each sound is represented as a vector of five multinomials, taking on two to eight values, where the first entry is its type (consonant or vowel) and the remaining four entries are as listed in Figure 2. We also encode word boundaries (denoted by $\#$) and dots (deletions/insertions) as extra types, with no additional features.

Consonant articulation		
M	**Manner**	plosive, fricative, glide, ...
P	**Place**	labial, dental, ..., velar, uvular
X	**Voiced**	– , +
S	**Secondary**	– , affricate, aspirate, ...
Vowel articulation		
V	**Vertical**	high—mid—low
H	**Horizontal**	front—center—back
R	**Rounding**	– , +
L	**Length**	1—5

Figure 2. Phonetic features for consonants and vowels.

We employ the MDL Principle [7] for model class selection and the MDL cost consists of two parts. First, we encode the model class $\mathcal{C}$, which is determined by a set of 18 decision trees, one for each feature (type plus four consonant and four vowel features) on both levels—source and target language. These trees query some *context* at each inner node, and their leaves provide the distribution to be used to encode the corresponding feature of a sound. More precisely the model (class) is allowed to query a fixed, finite a set of *candidate* contexts. A context is a triplet (L, P, F), where L is the level (source or target), P is a *position* relative to what we are currently encoding, and F is one of the possible features found at that position. An example of allowed candidate positions is given in Figure 3. In this setup, we have 2 levels $\times$ 8 positions $\times$ 2–6 features ≈ 80 candidate contexts, one of which defines an inner node of a feature tree. We can therefore encode each tree using one bit per node to indicate whether it is a leaf or not, plus about $\log 80$ bits for each inner node to specify the context on which it splits. For a model class $\mathcal{C}$, we need to encode all of its 18 trees in this way, the resulting total code-length we denote $L(\mathcal{C})$.

Context Positions	
I	itself, possibly dot
-P	previous position, possibly dot
–S	previous non-dot symbol
–K	previous consonant
–V	previous vowel
+S	previous or self non-dot symbol
+K	previous or self consonant
+V	previous or self vowel
...	*(other contexts possible)*

Figure 3. Context positions that a feature tree may query.

The second part of the code-length comes from encoding the aligned data using model class $\mathcal{C}$. We encode the feature in some fixed order, type first for it determines which other features need to be encoded. For each sound and each feature, we take a path from the root of the corresponding tree of $\mathcal{C}$ to a leaf, following at each inner node the branch that corresponds to the current context which is being queried. For example, when encoding feature $\mathbf{X}$ (voicedness) of a symbol σ in the source language we may arrive at a node given by $(L, P, F) = (target, -K, \mathbf{M})$ querying the manner of articulation of the previous consonant on the target level. This value (any manner of articulation or 'n/a' if there is no consonant on the target level between the current position and the beginning of the word) determines the edge we follow down the tree.

Each path from the root of a tree to a low-entropy leaf can be interpreted as as rule of phonetic correspondence. The path describes a contextual condition, the leaf gives the correspondence itself. High-entropy leaves represent variation that the model cannot explain.

In this way, all features of all symbols arrive at some node in the corresponding tree. We encode this data at each leaf independent of all other leaves using the normalized maximum likelihood (NML) distribution [8]. As the data at each leaf is multinomial, with cardinality $|F|$—the number of values feature F can take on—the corresponding code-length can be computed in linear time [9].

When $\mathcal{C} = \{\mathcal{T}_F^L\}$ consists of trees $\mathcal{T}_F^L$ for level L and feature F, and $\mathcal{D}$ is the aligned corpus such that $\mathcal{D}_{|L,F,\ell}$ is the portion arriving at a leaf $\ell \in \mathcal{T}_F^L$, then the overall code-length for $\mathcal{D}$ using $\mathcal{C}$ is

$$L(\mathcal{D}, \mathcal{C}) = L(\mathcal{C}) + \sum_L \sum_F \sum_\ell L_{NML}(\mathcal{D}_{|L,F,\ell}). \quad (1)$$

As implied, $L_{NML}(\mathcal{D}_{|L,F,\ell})$ is the multinomial stochastic complexity of the restricted data $\mathcal{D}_{|L,F,\ell}$. This code-length is the criterion to be minimized by the learning algorithm.

4. LEARNING

We start with an initial *random* alignment for each pair of words in the corpus. We then alternatively re-build the decision trees for all features on source and target levels as described below, and re-align all word pairs in the corpus using standard dynamic-programming, an analog procedure to the one described in [4]. Both of these operations decrease code-length. We continue until we reach convergence.

Given a complete alignment of the data, for each level L and feature F we need to build a decision tree. We

want to minimize the MDL criterion (1), the overall code-length. We do so in a greedy fashion by iteratively splitting the level-feature restricted data $\mathcal{D}_{|L,F}$ according to the cost-optimal decision (context to split upon). We start out by storing $\mathcal{D}_{|L,F}$ at the root node of the tree, e.g., for the voicedness feature **X** in Estonian (aligned to Finnish) we store data with counts:

+	801
-	821

In this example, there are 1622 occurrences of Estonian consonants in the data, 801 of which are voiced. The best split the algorithm found was on (Source, I, **X**), resulting in three new children. The data now splits according to this context into three subsets with counts:

+	
+	615
-	2

-	
+	135
-	764

n/a	
+	51
-	55

For each of these new nodes we split further, until no further drop in total code-length can be achieved. A split costs about $\log 80$ plus the number of decision branches in bits, the achieved gain is the drop in the sum of stochastic complexities at the leaves obtained by splitting the data.

5. EVALUATION

We present two views on evaluation: a *strict* view and an *intuitive* view. From a strictly information-theoretic point of view, a sufficient condition to claim that model (class) M_1 is better than M_2, is that M_1 yields better compression of the data. Figure 4 shows the absolute costs (in bits) for

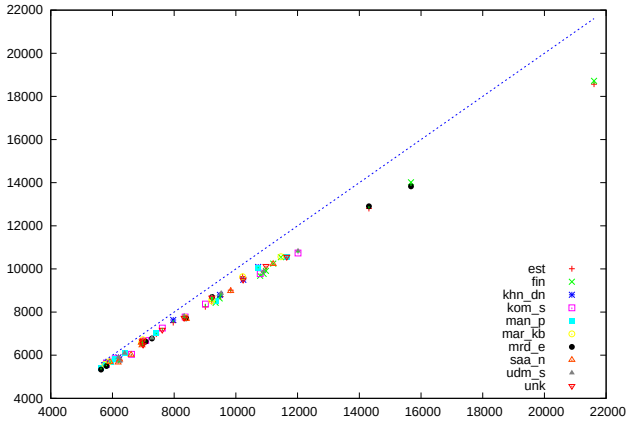

Figure 4. Comparison of code-lengths achieved by context model (Y-axis) and 1-1 baseline model (X-axis).

all language pairs[1]. The context model always has lower cost than the 1-1 baseline presented in [4]. In figure 5, we compare the context model against standard data compressors, Gzip and Bzip, as well as models from [4], tested on over 3200 Finnish-Estonian word pairs from SSA [3]. Gzip and Bzip need not encode any alignment, but neither can they exploit correspondence of sounds. These comparisons confirm that the new model finds more regularity in the data than the baseline model does, or an off-the-shelf data compressor, which has no knowledge that the words in the data are etymologically related.

[1]The labels appearing in the figures for the 10 Uralic languages used in the experiments are: est=Estonian, fin=Finnish, khn=Khanty, kom=Komi, man=Mansi, mar=Mari, mrd=Mordva, saa=Saami, udm=Udmurt, unk/ugr=Hungarian.

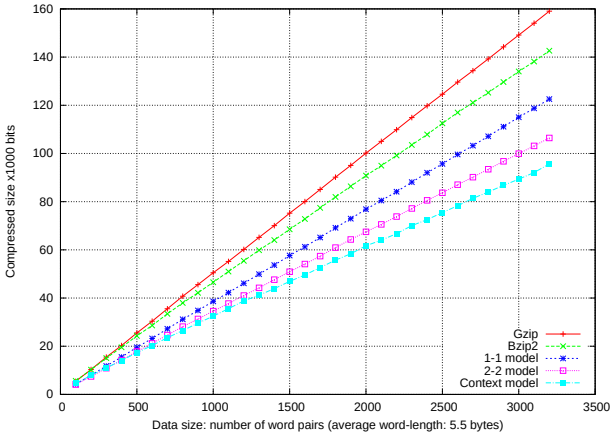

Figure 5. Comparison of compression power

For a more intuitive evaluation of the improvement in the model quality, we can compare the models by using them to *impute* unseen data. For a given model, and a language pair (L_1, L_2)—e.g., (Finnish, Estonian)—hold out one word pair, and train the model on the remaining data. Then show the model the hidden Finnish word and let it impute (i.e., guess) the corresponding Estonian. Imputation can be done for all models with a simple dynamic programming algorithm, very similar to the one used in the learning phase. Formally, given the hidden Finnish string, the imputation procedure selects from all possible Estonian strings the most probable Estonian string, given the model. Finally, we compute an edit distance (e.g., the Levenshtein distance) between the imputed string and the correct withheld Estonian word. We repeat this procedure for all word pairs in the (L_1, L_2) data set, sum the edit distances, and normalize by the total size (number of sounds) of the correct L_2 data—giving the *Normalized Edit Distance*: $NED(L_2|L_1, M)$ from L_1 to L_2, under model M.

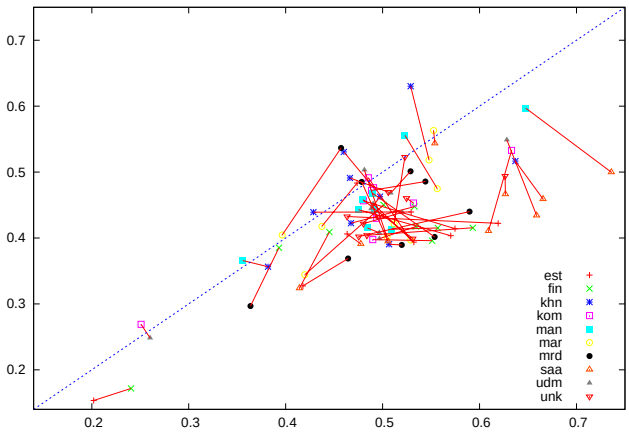

Figure 6. *Comparison of NED of context model (Y-axis) and "two-part 1-1" model (X-axis).*

The NED indicates how much regularity the model has captured. We use NED to compare models across all languages, Figure 6 compares the context model to the "two-part 1-1" model from [4]. Each of the $10 \cdot 9$ points is a directed comparison of the two models: the source language is indicated in the legend, and the target language is identified by the other endpoint of the segment on which the point lies. The further away a point is from the diagonal, the greater the advantage of one model over the other.

The context model always has lower cost than the baseline, and lower NED in 88% of the language pairs. This is an encouraging indication that optimizing the code length is a good approach—the models do *not* optimize NED directly, and yet the cost correlates with NED, which is a simple and intuitive measure of model quality.

A similar use of imputation was presented in [5] as a kind of cross-validation. However, the novel, normalized NED measure we introduce here provides yet another inter-language distance measure (similarly to how NCD was used in [4]). The NED (distances) can be used to make inferences about how far the languages are from each other, via algorithms for drawing phylogenetic trees. The pairwise NED scores were fed into the NeighborJoin algorithm, to produce the phylogeny shown in Fig. 7.

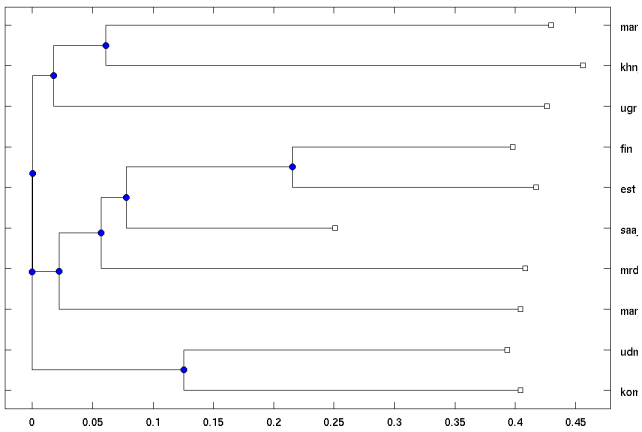

Figure 7. Finno-Ugric tree induced by imputation and normalized edit distances (via NeighborJoin)

To compare how far this is from a "gold-standard", we can use, for example, a distance measure for unrooted, leaf-labeled (URLL) trees found in [10]. The URLL distance between this tree and the tree shown in Fig. 1 is 0.12, which is quite small. Comparison with a tree in which Mari is not coupled with either Mordva or Permic—which is currently favored in the literature on Uralic linguistics—makes it a perfect match.

6. DISCUSSION AND FUTURE WORK

We have presented a feature-based context-aware MDL alignment method and compared it against earlier models, both in terms of compression cost and imputation power. Language distances induced by imputation allow building of phylogenies. The algorithm takes only an etymological data set as input, and requires no further assumptions. In this regard, it is as objective as possible, given the data (the data set itself, of course, may be highly subjective).

To our knowledge, this work represents a first attempt to capture *longer-range* context in etymological modeling, where prior work admitted minimum surrounding context for conditioning the edit rules or correspondences.

Acknowledgments

This research was supported by the Uralink Project of the Academy of Finland, and by the National Centre of Excellence "Algorithmic Data Analysis (ALGODAN)" of the Academy of Finland. Suvi Hiltunen implemented earlier versions of the models.

7. REFERENCES

[1] Sergei A. Starostin, "Tower of Babel: Etymological databases," http://newstar.rinet.ru/, 2005.

[2] Károly Rédei, *Uralisches etymologisches Wörterbuch*, Harrassowitz, Wiesbaden, 1988–1991.

[3] Erkki Itkonen and Ulla-Maija Kulonen, *Suomen Sanojen Alkuperä (The Origin of Finnish Words)*, Suomalaisen Kirjallisuuden Seura, Helsinki, Finland, 2000.

[4] Hannes Wettig, Suvi Hiltunen, and Roman Yangarber, "MDL-based Models for Alignment of Etymological Data," in *Proceedings of RANLP: the 8th Conference on Recent Advances in Natural Language Processing*, Hissar, Bulgaria, 2011.

[5] Alexandre Bouchard-Côté, Percy Liang, Thomas Griffiths, and Dan Klein, "A probabilistic approach to diachronic phonology," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, June 2007, pp. 887–896.

[6] David Hall and Dan Klein, "Large-scale cognate recovery," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

[7] Peter Grünwald, *The Minimum Description Length Principle*, MIT Press, 2007.

[8] Jorma Rissanen, "Fisher information and stochastic complexity," *IEEE Transactions on Information Theory*, vol. 42, no. 1, pp. 40–47, January 1996.

[9] Petri Kontkanen and Petri Myllymäki, "A linear-time algorithm for computing the multinomial stochastic complexity," *Information Processing Letters*, vol. 103, no. 6, pp. 227–233, 2007.

[10] D.F. Robinson and L.R. Foulds, "Comparison of phylogenetic trees," *Math. Biosci.*, vol. 53, pp. 131–147, 1981.