A Mixed Method Lemmatization Algorithm Using a Hierarchy of Linguistic Identities (HOLI)

Anton Karl Ingason[1], Sigrún Helgadóttir[2], Hrafn Loftsson[3], and Eiríkur Rögnvaldsson[1]

[1] Department of Icelandic, University of Iceland
Árnagarður v/Suðurgötu, 101 Reykjavik, Iceland
[2] The Árni Magnusson Institute for Icelandic Studies
Neshagi 16, 107 Reykjavik, Iceland
[3] School of Computer Science, Reykjavik University
Kringlan 1, 103 Reykjavik, Iceland
anton@akademia.is, sigruhel@hi.is, hrafn@ru.is, eirikur@hi.is

Abstract. We present a new mixed method lemmatizer for Icelandic, *Lemmald*, which achieves good performance by relying on IceTagger [1] for tagging and The Icelandic Frequency Dictionary [2] corpus for training. We combine the advantages of data-driven machine learning with linguistic insights to maximize performance. To achieve this, we make use of a novel approach: Hierarchy of Linguistic Identities (HOLI), which involves organizing features and feature structures for the machine learning based on linguistic knowledge. Accuracy of the lemmatization is further improved using an add-on which connects to the Database of Modern Icelandic Inflections [3]. Given correct tagging, our system lemmatizes Icelandic text with an accuracy of 99.55%. We believe our method can be fruitfully adapted to other morphologically rich languages.

Keywords: lemma, lemmatization, normalization, machine learning, BLARK, Icelandic, Lemmald, IceTagger.

1 Introduction

Lemmatization is the task of finding the base form – the *lemma* – of a given word form. The process is similar, while not identical, to the task of stemming which removes affixes from a word and returns the stem, the largest common part shared by morphologically related forms. Lemmatization and stemming are normalization techniques which serve the purpose of creating a connection between related words or word forms. Such a normalization is important in various natural language processing (NLP) applications, such as text classification and information extraction, because it brings out actual grammatical or semantic relations which are otherwise not accessible by the software (e.g. [4,5,6]).

In this paper, we describe *Lemmald*, a new lemmatizer for Icelandic, written in Java. Since Icelandic is a highly inflected language, one of the most important units in an Icelandic BLARK (Basic Language Resource Kit) [7] is an effective

A. Ranta, B. Nordström (Eds.): GoTAL 2008, LNAI 5221, pp. 205–216, 2008.

lemmatizer. Until now, the only available lemmatizer for Icelandic has been the language-independent CST Lemmatizer which has been trained for Icelandic [8]. A practical motivation for developing another lemmatizer for Icelandic is, for example, to be able to integrate it easily with other tools in the *IceNLP* toolkit which is currently being developed [9]. Furthermore, the existence of *Lemmald* gives the Icelandic NLP community the possibility of further improving the accuracy in lemmatization without having to rely on a language-independent lemmatizer. Unique aspects of the Icelandic language can thus by directly mirrored in the program code.

The tagset we use for Icelandic is the one developed for the Icelandic Frequency Dictionary corpus (IFD) [2]. It consists of about 700 different POS tags, where each character in the tag string corresponds to a single morphosyntactic category. The first character always marks the part of speech. Thus, the sentence *Hún mætti manninum* 'She met the man' will be tagged like this:

(1)

Hún	fpven
mætti	sfg3eþ
manninum	nkeþg

The meaning of the tags is as follows: **fpven**: pronoun (f) - personal (p) - feminine (v) - singular (e) - nominative (n); **sfg3eþ**: verb (s) - indicative (f) - active (g) - 3rd person (3) - singular (e) - past (þ); **nkeþg**: noun (n) - masculine (k) - singular (e) - dative (þ) - suffixed article (g).

In addition to dealing with a large tagset, an Icelandic NLP tool must address the fact that several word formation processes are very active in the language. Any given text is likely to contain a number of new compounds or derived forms which a data-driven solution has not encountered when trained using a corpus. To handle this, some kind of a compound analyzer is an important tool.

Lemmald uses a new algorithm for lemmatizing morphologically rich languages, combining data-driven machine learning methods and linguistic knowledge. The lemmatizer relies on three external NLP resources developed in previous projects. It uses the rule-based POS tagger *IceTagger* [1] for tagging its input and it is trained using the IFD corpus. Furthermore, *Lemmald* can optionally be run with the Database of Modern Icelandic Inflections (DMII) [3] as an add-on for improved results.

Our evaluation shows that, given correct tagging, *Lemmald* lemmatizes with an accuracy of 98.54%. Using the DMII as an add-on further improves the result to an accuracy of 99.55%. We consider this success an indication of the tool being ready to be used in practical situations for purposes of linguistic research or commercial software development.

The paper is organized as follows. In Sect. 2, we discuss related work concerned with normalizing text. Section 3 presents the external NLP resources used by *Lemmald*, and Sect. 4 describes some language specific issues when lemmatizing Icelandic. Section 5 describes our algorithm and in Sect. 6 we present an evalution of our system and the CST Lemmatizer. We conclude, in Sect. 7, with a summary.

2 Related Work

A frequently cited example of text normalization is the Porter Stemming Algorithm [10]. Development of the algorithm is motivated by the idea that “the performance of an IR system will be improved if [related] term groups [...] are conflated into a single term”. The Porter Stemmer removes suffixes from a word form until the “stem” is found. The stem may or may not be a linguistically “correct stem” but in most cases serves well the purpose of reducing the size and complexity of the data in the text it processes. For this purpose, the use of a stemmer instead of a lemmatizer is often desirable, because it can make a connection between a noun and a verb with the same stem which frequently reflects an actual semantic relation. However, the method of suffix stripping does not bring out such connections in irreglar inflection (e.g. *good*, *better*) – a job better suited for a lemmatizer (given that the lemmatizer can handle such irregularities). Actually, the advantages of both approaches can be combined to a considerable extent by running a lemmatizer first and subsequently stemming the lemmatized form.

The CST Lemmatizer [11] and the Euroling Stemmer [12] have been used for normalizing text in the Nordic languages, cf. [13]. Those two normalization systems represent the difference between stemming and lemmatization and also the other main distinction in software of this kind – that is the difference between hand-crafted and machine learned rules. The Euroling Stemmer uses only hand-crafted rules while the CST Lemmatizer uses only machine learned rules.

The method used by the CST Lemmatizer involves discovering suffix substitution rules by examining a tagged and lemmatized training corpus. When applying the rules to input data the rule with the longest suffix that matches the input word (and tag if present) is selected. The training phase is responsible for organizing the suffix substitution rules to maximize the probability of the longest matching suffix in the rule set, resulting in a correct lemma.

There are two main differences between our *Lemmald* and these techniques. Instead of focusing entirely on either hand-crafted rules or machine learning, we attempt to combine linguistic knowledge with machine learning. Then we attempt to use this combination to select from rules that apply minimally. By rules that apply minimally, we mean that we use the shortest suffixes that will map an input word to its lemma because *Lemmald* does not base its decisions on suffix length. In the evaluation of *Lemmald*, we compare our results with running the CST Lemmatizer on the same data as presented in Section 6.

3 External NLP Resources

3.1 Tagged Icelandic Corpus

Lemmald is designed to be trained on a tagged and lemmatized corpus. The only such corpus available for Icelandic is the IFD corpus. The IFD corpus contains about 590k tokens, where each token consists of a word form, a manually corrected POS (morphosyntactic) tag, and a lemma. This kind of data is well suited for training NLP tools, but the main weakness is that most of the texts in the corpus are literary works and thus the corpus may not represent other text categories as well. In terms of our lemmatizer it may perform slightly better if the input resembles literary texts.

3.2 IceTagger

Our lemmatizer can lemmatize text that has already been tagged, but in the common case of untagged texts, *Lemmald* uses IceTagger to tag the input before lemmatization is performed. Evaluations have indicated that IceTagger gives the correct tag about 91.5% of the time [1].

This relatively low tagging accuracy, compared to related languages, seems to limit the possibility of lemmatization which is based on the tag as well as the word form. However, the impact of incorrect tagging is quite low, since most tagging errors do not involve selecting the wrong word class (noun, verb, etc.), but rather incorrect subfeatures such as case, number, and gender in ambiguous word forms. Such an error does not affect lemmatization, because, for example, the lemma for a noun is the same irrespective of case. Indeed the accuracy in lemmatization is remarkably higher than the tagging accuracy. Accuracy when tagging Icelandic only with respect to word class has been measured as high as 98.14% [14] which is substantially higher than when including all subfeatures.

3.3 Database of Modern Icelandic Inflections

Although it is possible to achieve relatively good results using only data from the IFD corpus, a larger database of words can improve the results by filling in the gaps. Therefore, we have included the option of running *Lemmald* with an add-on which communicates with the DMII. Note that the DMII does not contain any frequency data and thus complements, rather than replaces, the IFD corpus.

The DMII is a huge database and its size causes some practical issues in terms of implementation and performance. For our purposes, we use a format where the data is contained within a table and each row consists of a set of a word form, a tag and a lemma. All combinations of grammatical categories for nouns, verbs and adjectives are included and in total there are over 5 million rows in the table. Note that other word classes are closed and are thus well covered by the IFD corpus. Our add-on is designed to communicate with any database management system which contains such a table and supports JDBC connections. In our development tests, we used MS SQL Server which worked quite efficently but frequent database queries will nevertheless affect the performance of any software. Thus, the improved lemmatization is traded for slower execution speeed when running the DMII add-on.

4 Language Specific Issues

When lemmatizing an unknown word in a language which uses suffixes for encoding grammatical distinction, it seems intuitive to search for the longest known ending of the unknown word. This method is indeed frequently used in lemmatization (e.g. [11]). However, applying longest match substitution blindly to Icelandic words can lead to mistakes.

Consider, for example, the word *götusópari* 'street sweeper' which is a compound made of the genitive of the noun *gata* 'street' and a noun derived from the verb *sópa*, using the affix *-ari* 'one who does what the verb expresses' (similar to the English *-er*). The masculine noun *götusópari* does not appear in the 590K word IFD corpus nor does the noun *sópari* 'sweeper'. On the other hand, the neuter noun *pari* (dative form of *par* 'pair') does exist in the corpus. Using longest match analysis for unknown words results in the incorrect lemma *götusópar*, while the correct result would be an unmodified *götusópari*. In *Lemmald*, we still use longest match analysis for cases where other methods fail, but first we attempt to analyze compounds using other more precise modules.

Examples like *götusópari* are particularly difficult to handle, because both the word itself and its second half, *sópari*, (the grammatical head) are unknown. Usually, compound analysis for unknown word forms is easier when both parts are known. However, it is not always enough to know both parts because of the problem of compound ambiguity. Consider, for example, the feminine plural noun *álfelgur* 'alumininum rims'. Even if our lemmatizer recognizes the parts *ál* 'aluminium' and *felgur* 'rims' it might mistake the *ál-felgur* compound for *álf-elgur* 'elf moose', because of compound ambiguity.

Compound analysis may also fail because of what we might call *partially unknown word parts.* This occurs when both parts of the compound are known as word forms, but the grammatical head (the rightmost part) does not exist in the corpus in a context where it fully agrees with the provided morphosyntactic tag. Maintaining an agreement between the input tag and decisions of the lemmatizer is important, because on most occasions, the occurrence of a partially unknown compound can be attributed to a minor distinction within the inner structure of the tag. Perhaps a particular noun has the same written form in the nominative case and the accusative case, but the corpus only contains the accusative form.

An example of an Icelandic compound which could possibly be partially unknown is *drengja-móður* 'mother of boys'. The compound is in no way unusual Icelandic but it is unlikely to appear in a corpus. The word form *drengja-móður* is identical in the accusative, dative and genetive case. Even if the first part of the compound *drengja* 'boys' was known and the second *móður* 'mother' also – a problem would rise if the input *móður* is in genitive case but is only known in accusative case in the corpus. This can of course happen irrespective of whether the genitive form is a part of a compound or not. Thus, a mechanism is needed to fall back to an agreement on, say, word class and gender, even if it is impossible to also confirm agreement on case using the limited corpus.

To resolve this, we present the idea of a *Hierarcy of Linguistic Identities* (HOLI) for Icelandic. It uses a simple linguistic insight to maintain a mostly data-driven machine learning approach when lemmatizing compounds, while falling back to nonperfect data in a linguistically sensible way when there are gaps in the training data.

5 Algorithm

5.1 Overview

We define the task of lemmatizing an Icelandic word as the one of implementing the function *getLemma(wordForm, tag)*. This definition implies that for a given word form and tag there exists one and only one correct lemma and it also leaves all issues of context sensitivity to the task of determining the tag. In this sense, our method is identical to the one of the CST Lemmatizer. The observation that an input of word form and tag corresponds to a unique lemma holds for almost all cases in Icelandic and the rare exceptions represent an insignificant part of lemmatization errors. Thus, the input to *Lemmald* consists of a wordform and a morposyntactic tag, whose format is described in Sect. 1. The tag can be obtained by using any POS tagger for Icelandic, but, as discussed in Sect. 3, we use IceTagger for this purpose.

Lemmald uses a mixed method approach to perform its task. The main method is the HOLI method mentioned in the previous section. However, further techniques are required to handle special cases, such as compound analysis, umlaut substitution and various systematic exceptions to Icelandic morphology. Finally, the add-on that connects to the DMII can be used for improved results. Units of functionality are organized into modules that can be turned on or off by adjusting configuration parameters of the program. The modules are as follows: (1) Hierarchy of Linguistic Identities Analysis, (2) Compound Analysis, (3), Umlaut substitution, (4) Post processing, (5) Database of Modern Icelandic Inflections Lookup.

5.2 Hierarchy of Linguistic Identities

It is a challenge for a data-driven NLP method to handle input that it does not recognize from its training data. When working with fully known data patterns is not an option, some kind of fallback to a more general method is inevitable and the goal of machine learning is of course to be able to apply learned patterns to new data. However, it may not be the best strategy to think of the problem exclusively from the point of view of machine learning, because the success of such an approach also depends on the features fed to the machine and the structure of those features. This is where linguistic insights can be important, as Manning has recently emphasized [15].

When dealing with the Icelandic tagset of about 700 different tags, data sparseness problems are bound to occur and handling them well is essential. Our approach to this task makes use of a HOLI. An example of a data sparseness gap which can be resolved by HOLI analysis is if a corpus does not contain a particular case of a noun. Consider, for example, the previously mentioned word form *móður*, which may be the accusative, dative or genitive case of the feminine noun *móðir* 'mother' (singular). This word can also be a masculine singular form of an adjective which means 'winded, out of breath'. The format of the morphosyntactic tag for an Icelandic noun has four characters; the first character 'n' stands for noun, the second character is for gender, the third is for number and the fourth is for case. The noun *móður* can thus have the following tags: *nveo*, *nveþ*, *nvee*, where the second letter stands for feminine and the final letter stands for accusative, dative and genitive case, respectively. The adjective *móður* has the tag *lkensf* which stands for adjective, masculine, singular, nominative, strong declension, and positive degree, respectively.

Let us imagine that the lemmatizer is asked to lemmatize the word *móður* with the tag *nvee* (noun, genitive), and while the genitive form of 'mother' is not present in the training data there are a few occurrences of the identical accusative and dative forms. If the tag was treated as one unit having no structure and the fallback mechanism would just pick the most frequent lemma for the word form according to the corpus, we might get the adjective form *móður* if the adjective happened to occur frequently in the training data. Therefore, we generate four levels of identities for <*móður,nvee*> from specific to general:

(2)

word	tag
móður	nvee
móður	nv
[any]	nvee
móður	[any]

Note that this particular hierarchy may not be the optimal representation of a noun, it is simply something we have found to work well for our purpose. We create an intermediate level of specificness for *feminine noun* (*nv*), but we do not use number and case for creating such identities – those are just reflected in the full tag string. The study of how it is best to construct hierarchies of linguistic features is a complex issue. An example of such work in linguistics is the feature tree in phonology (e.g. [16]).

When matching input words to machine learned rules, the lemmatizer goes for the most specific matching identity. Note that when making a decision based on specificness like here, we use strict domination of the most specific level relevant to the given input. A lower ranking identity has no significance if it is possible to base a decision on a higher ranking identity. In this sense, our model is similar to Optimality Theory [17]. In the case of the genitive *móður*, it would be the identity *feminine noun* resulting in correct lemmatization even if there were no useful clues for lemmatizing the word according to its genitive case.

During the training phase, a HOLI is generated for every pair of word form and tag encountered in the training data (the IFD corpus) along with a rule which correctly lemmatizes the given word. An example HOLI along with the corresponding lemmatization rules for <*móður,nveo*> is as follows:

(3)

word	tag	rule
móður	nveo	ur>ir
móður	nv	ur>ir
[any]	nveo	ur>ir
móður	[any]	ur>ir

The rule is the minimal suffix substitution needed to map the full word form to its lemma. In a second run through the corpus, the full hierarchy is again generated for each word and tested against all matching rules created in the previous run, in order to obtain a score for each combination of identity+rule depending on how often that combination results in a correct lemma. The score is recorded in a rule database along with the identity and the rule. This score is used to select a rule if more than one possible rule is available within a specificness level.

If the lemmatizer knows the above rules for *<móður,nveo>* but has not seen *<móður,nvee>* it can not use the most specific identity in that case. Then it must try a lower ranking one. The HOLI for *<móður,nvee>* is shown in (2). The most specific known identity is *<móður,nv>* and therefore the rule *ur>ir* is applied resulting in the correct lemma *móðir*. Without the intermediate level of specificness, there would have been a conflict between the noun form *móður* and the identical masculine adjective which might have given the rule *r>r*, resulting in a lemmatization error.

Using a HOLI, we can still rely on machine learning to perform most of the work and save time that would otherwise be spent on manually writing linguistic rules. The HOLI takes care of "coming up with" linguistic insights such as picking a pattern from a feminine noun instead of a masculine noun, or a pattern from a noun rather than an adjective where appropriate. This way we can combine some of the advantages of data-driven and linguistic rule-based NLP.

It is important to note that the key observation here is the general idea of combining linguistic structure with how the machine learns, not this particular implementation. A strictly machine learning motivated study would probably treat the word form and the tag as two features with no internal structure. Even if such an approach attempted to "machine learn" the internal structure of the tag it could not make use of the linguistic understanding of "specificness" we employ with little effort here. This sort of thinking provides opportunities for linguistics to contribute to NLP without switching from machine-learning to hand-crafted rules. Instead of choosing between the approaches, they are combined.

5.3 Compound Analysis

By checking for the existance of an identity of the most specific level (which is identical to a dictionary lookup), the compound analyzer determines if the input word is known. This happens before the HOLI analysis and if the word is known, there is no reason to attempt compound analysis. In contrast, an unknown word can go through up to three levels of compound analysis: strict analysis, loose analysis and longest match analysis which is attempted only if the previous methods fail.

Strict analysis requires that both parts of the compound are known and that the tag of the the latter part (the grammatical head) is known to exist for that word form. Loose analysis has the same requirements for the grammatical head, but tries to construct a well formed first part using a few known Icelandic derivation methods. If the word is still unknown and is longer than 6 letters an attempt is made to find the longest matching known ending while requiring that the first part has at least one vowel. Should the compound analysis result in a successfully split compound, the grammatical head is sent to HOLI analysis along with its tag.

Let us, for example, say that the input is the noun <*hestaskip,nheo*> 'horse ship'. Strict analysis then determines that while this is an unknown word it is probably a compound made of the parts *hesta-skip* since *hesta* is a known word and *skip* is known to have the tag <*nheo*>. Then the input to the HOLI module becomes <*skip,nheo*> and the compound analyzer makes sure that the first part *hesta* is added to the result before it is returned.

All of the above methods can of course fail to find a probable compound analysis of the input word as is supposed to happen if the word is not really a compound, but a (simple) unknown word. Then a HOLI analysis takes over.

Although the compound analyzer works in most cases, it is a module which can without a doubt be improved. The authors hope to develop an independent and powerful unit for this task in future research as a contribution to the Icelandic BLARK.

5.4 Umlaut Substitution, Post-Processing and DMII Lookup

Umlaut substitution is a known issue in the lemmatization of other Germanic languages (e.g. [18]). For common words, the automatically generated rules in the HOLI analysis take care of changing 'ö' in an inflected form to 'a' in the lemma, but to make sure that this happens in less common words as well, every rule which removes the umlaut trigger 'u' causes the umlaut substitution module to reverse its effect in an affected root if appropriate.

For example, if the rule *u>a* is applied to the noun *tösku* 'bag' (accusative, dative or genitive) the resulting lemma without umlaut substitution would be **töska*. The umlaut module of *Lemmald* corrects this by substituting the 'ö' in the root for 'a' giving the correct lemma *taska*.

A few systematic errors appear in the output of the lemmatizer due to irregularities in Icelandic morphology. Some of those, particularly the ones that result in consonant clusters which violate constraints of Icelandic phonology, are corrected using a list of substitutions which is applied after all other modules have finished their work. The program must be taught to perform u-epenthesis when its machine-learned rules result in word final consonant clusters like *-kr* replacing them with *-kur*.

As previously mentioned, the program can be configured to communicate with the DMII using an add-on. The format of the database we use consists of just over 5 million rows, each containing a word form and a tag along with the corresponding lemma. If turned on, this module is run before the HOLI analysis. This improves precision of the lemmatization while slowing it down.

6 Evaluation

To evaluate the performance of *Lemmald*, and the effect of the modules it uses, we ran a 10-fold cross validation test on the IFD corpus where the size of each training set was about 530k tokens and each test set contained about 60k tokens. We used the word forms and the manually corrected morphosyntactic tags from the corpus and measured the success of *Lemmald* in finding the correct lemma for each token. We also trained the CST Lemmatizer using the IFD corpus and performed an identical evaluation – with and without tags in the input. The results are presented in Table 1, where mean accuracy is shown. In the first row the success for the HOLI method without any additional modules is shown and in each of the following rows one module is added to improve accuracy. The last row contains the results of our evaluation of the CST Lemmatizer. Note that, while assuming correct tagging is useful for evaluating different lemmatization methods, real world results will in most cases rely on machine tagged text which negatively affects accuracy. A preliminary test with one test set containing approximately 10,000 words was performed using *Lemmald*. This test showed a drop in accuracy of about 1.5% between lemmatizing correctly tagged text and a text tagged with IceTagger. The accuracy of the lemmatization is still much higher than the accuracy of the machine tagging, because, as pointed out earlier, most tagging errors do not affect lemmatization.

Table 1. Results

Lemmald	Tagged Input	Untagged Input
Basic (HOLI only)	97.85%	
+ Compound Analysis	98.38%	
+ Umlaut Substitution	98.42%	
+ Post processing	98.54%	
+ DMII	99.55%	
CST Lemmatizer		
	98.99%	93.15%

The CST Lemmatizer trained on the IFD Corpus reaches 98.99% accuracy when applied to correctly tagged text. A comparable number for *Lemmald* is 98.54%, which is obtained by omitting the use of the DMII. The difference is statistically significant ($\alpha < 0.001$). However, the difference between the best result for *Lemmald*, 99.55% obtained when the DMII is used, is significantly higher than the result with the CST lemmatizer (98.99%). Adding the DMII should have the same effect on the CST Lemmatizer resulting in an even higher accuracy. However, the above comparision has already confirmed that the CST Lemmatizer performs better than *Lemmald* when trained on the same data so we have not implemented such an evalution setting. Instead, we focus on measuring the effect of each specialized module.

Taking a closer look at the *Lemmald* column in Table 1, we can see that every language specific module contributes to the accuracy of the lemmatizer. This clearly shows that addressing language specific issues does matter for the performance. The language independent aspects of *Lemmald* are still a little behind the CST Lemmatizer. The reason for this is that within a level of specificness in the HOLI, *Lemmald* uses a very primitive way of choosing between rules. In such a situation there is no linguistic evidence to base the decision on and a very simple score mechanism is employed. By improving the decision making on this level, we believe the combination of our linguistically inspired method and the powerful tools of data-driven methods can result in an Icelandic lemmatizer which outperforms both systems evaluated here. Currently, there are examples of each lemmatizer failing in a situation where the other succeeds. The strenghts of the HOLI method in dealing with data sparseness in a large tagset is in many cases successful, but in other cases when there is ambiguity within a specificness level the method fails. We intend to improve our system in a future version so that it covers most or all cases that can be learned in a language independent way and goes beyond that when used with the language specific modules. Thus, we believe that future versions of *Lemmald* (without using DMII) will outperform the CST Lemmatizer and the former could therefore be used in favour of the latter. Additionally, as mentioned in Sect. 1, *Lemmald* allows for an easy integration into the IceNLP toolkit, because both units are implemented in Java.

7 Conclusion

We have shown that the combination of linguistic knowledge with data-driven machine learning can resolve issues that are difficult to handle when using one of the approaches exclusively. Machine learning is essential to save the time otherwise needed for hand-crafting linguistic rules. However, using linguistics to determine features and feature structures can combine the advantges of both approaches. Our way to achieve such combination in lemmatization is based on a Hierarchy of Linguistic Identities. More important than this particular implementation of such a HOLI, is the idea that a similar approach can be used in a number of NLP tasks. We believe it is important to move away from viewing the field as "just a branch of applied machine learning" [15] and towards the strenghts of a truly cross disciplinary field. NLP may have "caught up with what statisticians and machine learning people have discovered in 400 years" [15] but now it is time to catch up with the knowledge of linguistics.

Lemmald is ready to be used in combination with other Icelandic BLARK units in various practical situations in linguistic reasearch and commercial software development. However, we will continue to improve the system and aim to increase the lemmatization accuracy in future versions. This will involve more intelligent machine learning to deal with cases where our method fails and development of a more advanced compound analyser.

References

1. Loftsson, H.: Tagging Icelandic text: A linguistic rule-based approach. Nordic Journal of Linguistics 31(1), 47–72 (2008)
2. Pind, J., Magnússon, F., Briem, S.: [The Icelandic Frequency Dictionary]. The Institute of Lexicography, University of Iceland, Reykjavik (1991)

3. Bjarnadóttir, K.: Modern Icelandic Inflections. In: Holmboe, H. (ed.) Nordisk Sprogteknologi 2005. Museum Tusculanums Forlag, Copenhagen (2005)
4. Korenius, T., Laurikkala, J., Järvelin, K., Juhola, M.: Stemming and lemmatization in the clustering of finnish text documents. In: CIKM 2004: Proceedings of the thirteenth ACM international conference on Information and knowledge management, pp. 625–633. ACM, New York (2004)
5. Braschler, B., Ripplinger, B.: How Effective is Stemming and Decompounding for German Text Retrieval? Information Retrieval 7(3-4), 291–316 (2004)
6. Airio, E.: Word normalization and decompounding in mono- and bilingual IR. Information Retrieval 9(3), 249–271 (2006)
7. Krauwer, S.: The Basic Language Resource Kit (BLARK) as the First Milestone for the Language Resources Roadmap. SPECOM-2003, Moscow, Russia, Accessed 01.04.2008 (2003), `http://www.elsnet.org/dox/krauwer-specom2003.pdf`
8. Cassata, F.: Automatic thesaurus extraction for Icelandic. BSc Final Project, Department of Computer Science, Reykjavik University (2007)
9. Loftsson, H., Rögnvaldsson, E.: IceNLP: A Natural Language Processing Toolkit for Icelandic. In: Proceedings of Interspeech 2007, Special Session: Speech and language technology for less-resourced languages, Antwerp, Belgium (2007)
10. Porter, M.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
11. Jongejan, B., Haltrup, D.: The CST Lemmatiser. Center for Sprogteknologi, University of Copenhagen version 2.9 (2005)
12. Carlberger, J., Dalianis, H., Hassel, M., Knutsson, O.: Improving precision in information retrieval for Swedish using stemming. In: Proceedings of NODALIDA 2001 – 13th Nordic conference on computational linguistics (2001)
13. Dalianis, H., Jongejan, B.: Hand-crafted versus Machine-learned Inflectional Rules: The Euroling-SiteSeeker Stemmer and CST's Lemmatiser. In: LREC 2006: Proceeding of the International Conference on Language Resources and Evaluation (2006)
14. Helgadóttir, S.: Testing Data-Driven Learning Algorithms for PoS Tagging of Icelandic. In: Holmboe, H. (ed.) Nordisk Sprogteknologi 2004. Museum Tusculanums Forlag, Copenhagen (2005)
15. Manning, C.: Focusing on Linguistic Representations [abstract]. In: The Natural Language and Speech Processing Colloquium, Stanford, January 19 (2005)
16. Kenstowicz, M.: Phonology in Generative Grammar (Blackwell Textbooks in Linguistics). Blackwell Publishers, Malden (1993)
17. Prince, A., Smolensky, P.: Optimality Theory: Constraint Interaction in Generative Grammar. Manuscript, Rutgers University and University of Colorado at Boulder. ROA [ROA #537] (1993/2002), `http://roa.rutgers.edu/`
18. Lezius, W., Rapp, R., Wettler, M.: A freely available Morphological Analyzer, Disambiguator, and Context Sensitive Lemmatizer for German. In: Proceedings of the COLING-ACL, pp. 743–747 (1998)