Neural Models for Predicting Celtic Mutations

Kevin P. Scannell
Department of Computer Science
Saint Louis University
St. Louis, Missouri, USA 63103
kscanne@gmail.com

Abstract

The Celtic languages share a common linguistic phenomenon known as *initial mutations*; these consist of pronunciation and spelling changes that occur at the beginning of some words, triggered in certain semantic or syntactic contexts. Initial mutations occur quite frequently and all non-trivial NLP systems for the Celtic languages must learn to handle them properly. In this paper we describe and evaluate neural network models for predicting mutations in two of the six Celtic languages: Irish and Scottish Gaelic. We also discuss applications of these models to grammatical error detection and language modeling.

Keywords: Celtic languages, initial mutations, Irish, Scottish Gaelic, language modeling, neural networks

1. Introduction

The Insular Celtic language family consists of two branches, the Goidelic (or Q-Celtic) branch, comprised of Irish, Manx, and Scottish Gaelic, and the Brythonic (or P-Celtic) branch, comprised of Welsh, Breton, and Cornish. All six languages are under-resourced in terms of language technology, although substantial progress has been made in recent years, especially for Irish and Welsh (Judge et al., 2012; Evas, 2013).

The Celtic languages share a linguistic phenomenon known as *initial mutations*. These are pronunciation and spelling changes that occur at the beginning of certain words based on the grammatical context. For example, the Irish word *bád* ('a boat') undergoes an initial mutation known as *lenition* when preceded by the first person singular possessive adjective *mo*, hence *mo bhád* ('my boat'). Each Celtic language has multiple mutation types, each one governed by sometimes-complicated rules that can be challenging for learners and native speakers alike. Initial mutations are quite common (occurring, for example, in about 15% of tokens in typical Irish corpora), and so all NLP technologies for the Celtic languages must handle them correctly.

Our goal in this paper is to describe and evaluate several neural network models for predicting Celtic mutations. We restrict ourselves to Irish and Scottish Gaelic for reasons we will make clear in §2.2 below.

We have two primary applications in mind for this work. First, mutation errors are among the most common made by learners of the Celtic languages, and a sufficiently accurate predictive model can be used as part of a system for detecting and correcting grammatical errors. We address this application in §4.1. The second application is to language modeling, the idea being to separately predict the probability of a demutated token given its history followed by the probability of a given mutation on that token. We show in §4.2 that the resulting factored language model (Bilmes and Kirchhoff, 2003) yields a decrease in perplexity over a baseline 5-gram model. This language model can in turn be incorporated into end-user technologies like machine translation engines, reducing the number of mutation errors output by such systems.

The current research landscape in NLP is dominated by work on English, and algorithms or architectures that give state-of-the-art results on English are often applied without modification to other languages. We hope that the results of this paper show that one can improve upon language-independent approaches by incorporating linguistic knowledge specific to a given language or language family.

The outline of the remainder of the paper is as follows. We begin in §2 with an overview of the initial mutations that occur in Irish and Scottish Gaelic, and we explore the information-theoretic content of the mutation system. In §3 we define and evaluate our neural network models for predicting mutations. The final section §4 discusses the applications of these models to Irish grammar checking and language modeling.

2. Celtic Initial Mutations

2.1. Definitions and Examples

We begin with descriptions of the initial mutations that occur in Irish and Scottish Gaelic: lenition, t-prothesis, h-prothesis, and eclipsis.

- **Lenition** is a "softening" of the initial consonants *b*, *c*, *d*, *f*, *g*, *m*, *p*, *s*, and *t*, that occurs in both languages in certain contexts. It is indicated in the modern orthographies by the insertion of an *h* after the initial consonant. For example, adjectives following a feminine noun are lenited: *beag* ('small'), but *bean bheag* ('small woman'). (This example works in both languages).

- **T-prothesis**. In both languages a *t-* is prefixed to a masculine noun beginning with a vowel in the nominative singular when preceded by the definite article. There is a similar (but distinct) phenomenon triggered by the definite article that occurs for some nouns beginning with *s*. This is represented as a prefixed *t-* in Scottish Gaelic orthography, so *slat* ('a stick') becomes *an t-slat* ('the stick'), but is written without the hyphen in Irish (*an tslat*). For the purposes of our models these are treated as a single type of mutation.

- **H-prothesis**. An *h* (modern Irish) or *h-* (Scottish Gaelic and older Irish orthographies) is prefixed to some words having an initial vowel, e.g. nouns preceded by the third person feminine possessive adjective *a* ('her'), hence *aisling* ('dream') becomes *a haisling* ('her dream', Irish), or *a h-aisling* (Scottish Gaelic).
- **Eclipsis**. Roughly speaking, eclipsis causes voiceless stops to become voiced and voiced stops to become nasal. The full set of orthographic changes is given in the following section as part of Algorithm 1. Eclipsis occurs in both languages but is usually not realized orthographically as an initial mutation in Scottish Gaelic. One example that does occur in both languages is eclipsis of an initial vowel after the first person plural possessive adjective, so *athair* ('father') becomes *ár n-athair* ('our father', Irish) and *ar n-athair* (Scottish Gaelic).

We define a set of labels $\mathcal{M} = \{\mathbf{L}, \mathbf{T}, \mathbf{H}, \mathbf{E}, \mathbf{N}\}$ where $\mathbf{L}$, $\mathbf{T}$, $\mathbf{H}$, and $\mathbf{E}$ correspond to the four mutations above, and the label $\mathbf{N}$ is used for tokens having no mutation.

There are dozens of rules governing exactly when each mutation occurs, and we make no attempt to cover them all here, instead referring the interested reader to (Tithe an Oireachtais, 2016) and (Bauer, 2011) for Irish and Scottish Gaelic respectively. The rules can be quite challenging for language learners, and there is even significant variation in how the mutations are used among native speakers, based primarily on dialect.

A rule-based system for predicting Irish mutations has been implemented in previous work of the author.[1] But this approach relies at minimum on a rich lexicon and accurate part-of-speech tagging, and fails to implement certain rules that would require deeper syntactic or semantic analysis. The resulting system suffers from the brittleness that plagues many rule-based NLP systems. The pure machine learning approach we propose in this paper aims to overcome these shortcomings.

2.2. Orthographic Transparency and Demutation

All of the initial mutations in Scottish Gaelic are orthographically transparent, by which we mean that they can be trivially and algorithmically removed whenever they occur. With a single exception that we will describe momentarily, the same is true for Irish. This means that we can produce unlimited amounts of training data labeled with the correct mutations for either language, starting from plain text corpora. This is in contrast with the other four Celtic languages where mutations are generally *not* algorithmically removable,[2] and so the approach of this paper does not apply directly to Manx Gaelic, Cornish, Breton, or Welsh.

For Irish, lenition, eclipsis, and t-prothesis can be removed algorithmically, but h-prothesis cannot be, because, unlike Scottish Gaelic, it is written without a hyphen in the standard orthography. Therefore, without making use of a comprehensive lexicon one cannot be certain if a given initial *h* represents h-prothesis or if it is instead an integral part of the word, e.g. *hidrigin* ('hydrogen'). Even a dictionary-based approached is doomed because of ambiguities like *aiste* ('an essay') vs. *haiste* ('a hatch'), and because of new or non-standard words that will always be missing from a dictionary. Instead, we will take a brute force approach and define every initial *h* in Irish to be an example of h-prothesis.

Let $\mathcal{V}$ be the set of legal tokens. In our experiments we will assume all tokens have been lowercased which simplifies the model and the rules below. The goal of Algorithm 1 is to define two functions; the first $\sigma : \mathcal{V} \to \mathcal{V}$ strips mutations and the second $\mu : \mathcal{V} \to \mathcal{M}$ maps a token to its mutation label.

Algorithm 1. Given a token w, this algorithm returns the demutated token $\sigma(w)$ and the mutation label $\mu(w) \in \mathcal{M}$.

1. If a token w begins with *bhf*, set $\mu(w) = \mathbf{E}$, and remove the *bh* to obtain $\sigma(w)$.
2. Otherwise, if the second letter of a token w is an *h*, set $\mu(w) = \mathbf{L}$, and remove the *h* to obtain $\sigma(w)$.
3. Otherwise, if a token w begins with *h-*, set $\mu(w) = \mathbf{H}$, and remove the *h-* to obtain $\sigma(w)$.
4. (Irish only) Otherwise, if a token w begins with *h*, set $\mu(w) = \mathbf{H}$, and remove the *h* to obtain $\sigma(w)$.
5. Otherwise, if a token w begins with *t-*, set $\mu(w) = \mathbf{T}$, and remove the *t-* to obtain $\sigma(w)$.
6. Otherwise, if a token w begins with *ts*, set $\mu(w) = \mathbf{T}$, and remove the initial *t* to obtain $\sigma(w)$.
7. Otherwise, if a token w begins with *n-*, set $\mu(w) = \mathbf{E}$, and remove the *n-* to obtain $\sigma(w)$.
8. (Irish only) Otherwise, if a token w begins with *mb*, *gc*, *nd*, *ng*, *bp*, or *dt*, set $\mu(w) = \mathbf{E}$, and remove the first letter to obtain $\sigma(w)$.
9. Otherwise, set $\mu(w) = \mathbf{N}$ and $\sigma(w) = w$.

Note that we have defined slightly different functions for the two languages, but in what follows we will abuse notation and write them simply as σ and μ, understanding that the additional rules for Irish will be applied only in the Irish experiments.

Since we are trying to explore the learnability of the mutation system from raw text, it is important to note that neither function definition encodes any information about the lexicon or the grammatical context in which these mutations occur; they are simply defined in terms of the orthographic changes in question. In the case of lenition, for example, we do not even encode the knowledge that lenition only applies to consonants. This means that σ and μ sometimes do the "wrong thing" linguistically, as in the case of Irish h-prothesis noted above ($\sigma(hidrigin) = idrigin$), or when applied to English words embedded in otherwise Gaelic

[1] See `https://cadhan.com/gramadoir/`.

[2] Consider for example the surface form *vea* in Manx Gaelic, which can be a lenited form of the noun *bea* ('life') as in *my vea* ('my life'), or else an eclipsed form of *fea* ('quiet, rest') as in *gow-jee nyn vea* ('take your (pl.) rest', cf. Matthew 14:41).

texts, e.g $\sigma(Chaucer) = Caucer$ or $\mu(tsunami) = \mathbf{T}$. There are also more subtle cases one could quibble over, for example lenited Irish words like *cheana* or *chugat* for which the unlenited form does not exist in the lexicon. Again, all of this is fine for our purposes: we have simply defined a task in which our labels differ in rare cases from the linguistically-correct ones. In theory, this complicates the learning process since examples like these can occur in contexts where one would not expect to see the corresponding mutation. But as we will see in the sections that follow, there was no practical impact on the neural networks we trained; they simply learned to predict, for example, a (very) high probability for the label **H** given a "demutated" token like *idrigin*, or for the label **L** given examples like *ceana*, *cugat*, or *Caucer*.

2.3. Mutations as Low-Entropy Features

Celtic mutations carry very little information. Informally speaking, if one were to remove all of the mutations from a text, a native speaker would be able to restore almost all of them correctly and unambiguously. One of our goals in this section is to make this notion more precise, and assign a numerical value (in units of bits per token) to the information content of initial mutations.

We will write P for a model that produces the probability of a mutation given the demutated target word and preceding context. More precisely, let $w_1, \ldots, w_N$ be a sequence of tokens, and let $\mathbf{m} \in \mathcal{M}$. For $k \in \{1, \ldots, N\}$, we write $P(\mathbf{m}|\sigma(w_1)\ldots\sigma(w_k))$ for the probability that $\mu(w_k) = \mathbf{m}$ given the demutated token $\sigma(w_k)$ and the history $\sigma(w_1)\ldots\sigma(w_{k-1})$. Conditioning our predictions on *demutated* words to the *left* of the target word will allow us to incorporate P into a full language model; see §4.2. for details.

We write the average log loss Λ of this model as

$$\Lambda = -\frac{1}{N}\sum_{i=1}^{N} \log_2 P(\mu(w_i)|\sigma(w_1)\ldots\sigma(w_i)) \quad (1)$$

with units of bits per token. This gives an estimate of the information content of initial mutations based on the given model P and test corpus $w_1, \ldots, w_N$.

Our claim is that Celtic mutations are "low-entropy" linguistic features, by which we mean that the quantity Λ will be close to zero given a sufficiently accurate model P. We will demonstrate this experimentally in the next section. This claim will come as no surprise to readers familiar with the rules for Irish and Scottish Gaelic initial mutations, since the mutations are often completely determined by the surrounding context, often the previous one or two words along with the initial letter of the target word.

There are, however, some well-known exceptions where mutations appear to carry important information. One example is the Irish third person possessive *a* which can mean "his", "her", or "their", with the correct sense sometimes being determined by the mutation on the possessed noun: *a bád* ('her boat', no mutation), *a bhád* ('his boat', lenited), *a mbád* ('their boat', eclipsed). Of course in many cases the correct sense can also be reliably guessed from the surrounding context (necessarily so in cases where the possessed noun does not admit an initial mutation), but sometimes this is challenging, especially for sentences in isolation, and the mutation does convey non-trivial information. Another well-known example is a difference in mutations that occurs between Irish dialects in the dative case. In the Ulster dialect, speakers usually lenite a noun in the dative, e.g. *ar an bhád* ('on the boat'), while in the other major dialects the noun is eclipsed (*ar an mbád*) and so the mutation conveys information about the dialect of the speaker or writer. We will return to the question of which mutations carry the most information in a data-driven way in §3.3.

3. Neural Networks for Mutation Prediction

3.1. Design and Implementation

The goal of this section is to define a number of models which predict probabilities of initial mutations. All models are evaluated according to their (base 2) log loss Λ as defined in Equation 1 above. Because we plan to incorporate these into full language models, it is important that the predictions be conditioned only on the demutated target word and any preceding words. See §4.2 below for further details.

Because five-fold classification problems evaluated via log loss are not particularly common in the literature, we will define and evaluate a few very simple baselines to help frame the problem. For each model, we assume we have seen a history of demutated tokens $\sigma(w_1)\ldots\sigma(w_k)$ and want to predict the probability that mutation $\mathbf{m} \in \mathcal{M}$ occurs on w_k. We denote the tokens in the training corpus by $t_1 \ldots t_N$.

- **Label priors**. The unmutated label **N** is by far the most common, so we do reasonably well by simply assigning to every token the prior probability of $\mathbf{m}$ as seen in training:

$$P(\mathbf{m}|\sigma(w_1)\ldots\sigma(w_k)) = \frac{1}{N}\sum_{\mu(t_i)=\mathbf{m}} 1$$

- **First letter**. Certain initial letters are incompatible with certain mutations. We take this into account in this baseline by assigning the maximal likelihood probability of $\mathbf{m}$ as estimated from training tokens t_i such that $\sigma(t_i)$ and $\sigma(w_k)$ have the same first character.

- **Unigram model**. This baseline assigns the probability

$$P(\mathbf{m}|\sigma(w_1)\ldots\sigma(w_k)) = \frac{\sum_{\mu(t_i)=\mathbf{m},\sigma(t_i)=\sigma(w_k)} 1}{\sum_{\sigma(t_i)=\sigma(w_k)} 1},$$

  but smoothed using add-α smoothing with the parameter α tuned on the development set.

- **Trigram model**. This model is analogous to the unigram but computes the maximal likelihood estimate of $\mathbf{m}$ based on trigrams (t_{i-2}, t_{i-1}, t_i) seen in training such that $(\sigma(t_{i-2}), \sigma(t_{i-1}), \sigma(t_i)) = (\sigma(w_{k-2}), \sigma(w_{k-1}), \sigma(w_k))$. In case of zero counts, we back off to a bigram estimate, and then to the unigram model above.

The trigram model gives a reasonably strong baseline, which is not surprising given the fact that mutations can often be predicted from the previous word or two. Improving the trigram to be competitive with our best neural models would require a more sophisticated backoff strategy along the lines of the generalized parallel backoff proposed in (Bilmes and Kirchhoff, 2003), and perhaps incorporating some linguistic knowledge. To see this, consider a prepositional phrase like the Irish *ar an mbád* ('on the boat') with eclipsis on the noun *bád* ('boat'). Were this trigram not seen in training, a naive backoff strategy would estimate the probability of eclipsis by backing off to the bigram *an bád*, and for this and most other nouns the counts would be dominated by nominative examples where eclipsis is highly unlikely.[3]

Next we will describe our three neural network models, beginning with definitions of the layers that are used in more than one model.

All three models use character embeddings; for this, we fixed a vocabulary $\mathcal{C}$ consisting of the most common 32 characters seen in training, and the models learn an embedding $\chi : \mathcal{C} \to \mathbb{R}^{10}$.

The second and third models make use of a trainable token embedding layer. Here we fix a "demutated vocabulary" $\mathcal{V}_\sigma$ consisting of the 100,000 most common tokens in training after removing mutations, and then this layer learns a mapping $\psi : \mathcal{V}_\sigma \to \mathbb{R}^{200}$.

The latter two models also make use of a character-level bidirectional LSTM (Graves and Schmidhuber, 2005) which provides a second embedding of each input token as a fixed-length vector, in this case one that hopefully captures internal orthographic or morphological features relevant to the mutation system. The character embedding χ is used to convert the characters in a demutated token into a sequence of vectors in $\mathbb{R}^{10}$ which is in turn input into a BiLSTM layer with 75 cells in each direction. The two unidirectional outputs are concatenated, defining a mapping $\beta : \mathcal{V}_\sigma \to \mathbb{R}^{150}$.

The top two (output) layers are the same across all three models: first, a dense layer with 100 cells and reLU activation, followed by a softmax layer that outputs a probability distribution over the set of five labels $\mathcal{M}$.

With this notation established, we define our three neural networks as follows:

- **Character LSTM**. In this model, we rejoin the full token history $\sigma(w_1), \ldots, \sigma(w_k)$ into one long string separated by spaces, and then extract the final 20 characters of the resulting string. The character embedding χ is then applied to produce a sequence of 20 vectors in $\mathbb{R}^{10}$ which are passed into an LSTM with 150 cells and recurrent dropout of 0.4. The final state vector output by the LSTM is fed into the dense and softmax output layers where the loss is computed.

- **Trigram and character BiLSTM**. Since mutations are usually triggered by one or two preceding words, we wanted to define a strong baseline that only considers the target token $\sigma(w_k)$ (for which we are predicting the mutation) and the two previous tokens $\sigma(w_{k-2})$ and $\sigma(w_{k-1})$. For $i \in \{0, 1, 2\}$ we embed $\sigma(w_{k-i})$ by concatenating the token embedding $\psi(\sigma(w_{k-i}))$ and the BiLSTM embedding $\beta(\sigma(w_{k-i}))$. The three resulting vectors are concatenated and fed directly into the dense and softmax output layers.

- **Token LSTM and character BiLSTM**. The idea here is to use an LSTM to encode a longer token history as a fixed-length vector, hopefully enabling the model to make better predictions in subtle cases (e.g. words preceded by third person possessive adjectives where anaphora resolution is needed). We fix a window size of L tokens. For $i \in \{0, \ldots, L-1\}$ we again embed $\sigma(w_{k-i})$ by concatenating the token embedding $\psi(\sigma(w_{k-i}))$ and the BiLSTM embedding $\beta(\sigma(w_{k-i}))$. The resulting sequence of L vectors is input into an LSTM with 500 cells and recurrent dropout of 0.25. The final state vector output by the LSTM is fed into the dense and softmax output layers; see Figure 1.

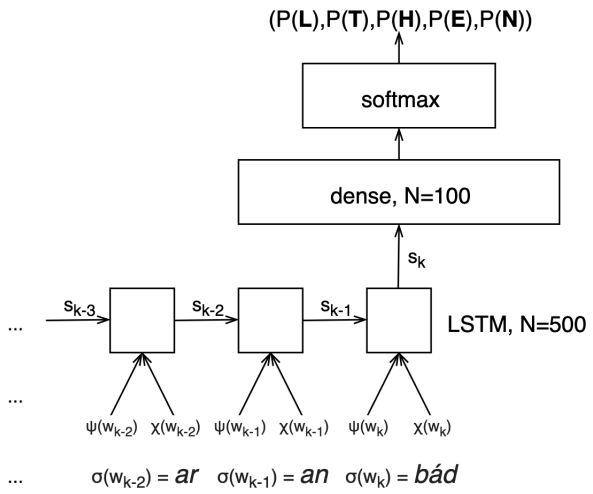

Figure 1: Architecture of the Token LSTM and character BiLSTM model

3.2. Training and Evaluation

The training, development, and test corpora for both languages were assembled by crawling the web (Scannell, 2007). The corpora are sentence-shuffled,[4] tokenized, and then lowercased.

It is important to take note of a subtlety when lowercasing Irish and Scottish Gaelic that involves initial mutations. When t-prothesis or eclipsis occurs with a capitalized vowel-initial word in Irish, the prefixed letter is written without a hyphen, e.g. *ár nAthair* ('our Father').

[3] We conjecture that this is the cause of many of the mutation errors seen in the output of machine translation engines that use n-gram language models.

[4] By shuffling sentences, we greatly increase the size of the corpora that can be freely distributed, with the tradeoff that it becomes impossible for the model to learn contextual clues across sentences, as is sometimes required for correct prediction.

Therefore, when lowercasing, the hyphen must be reinserted: *ár n-athair* and not *ár nathair* which would mean something completely different ('our snake'). We do the same thing when lowercasing Scottish Gaelic as well (where the hyphen is sometimes omitted), and additionally for h-prothesis, so *Pàrlamaid na hAlba* lowercases to *pàrlamaid na h-alba*.

The Scottish Gaelic training corpus contains 8 million tokens, and the development and test sets have 500k tokens each. There is substantially more Irish than Scottish Gaelic available online and so the corpora are much bigger: 50 million tokens of Irish for training, and 1 million tokens each for development and testing.

In performing error analysis on the Irish model (see §3.3), it became clear that a significant portion of the loss exhibited by our models came from grammatical errors in the test corpus rather than flaws in the model. For this reason, we created a second test set from a corpus of well-edited articles from the online Irish news service Tuairisc.ie. The full corpus consists of about 7.7 million tokens from which we extracted a 1 million token test set, disjoint from the training corpus, which was shuffled, tokenized, and lowercased as above.

The neural network models were implemented in TensorFlow (Abadi et al., 2016) and each was trained for 20 epochs, checkpointing and saving the models with smallest loss on the development set. The test losses for all models are reported in Table 1 in units of bits per token, and columns are labeled with ISO 639-1 language codes ("ga" for Irish and "gd" for Scottish Gaelic).[5]

Model	Test (ga)	Clean (ga)	Test (gd)
Label priors	0.75917	0.79581	0.66670
First letter	0.52064	0.54242	0.40187
Unigram model	0.40571	0.39485	0.31835
Trigram model	0.10710	0.07995	0.10205
Char LSTM	0.10336	0.08531	0.08598
3-gram+BiLSTM	0.08051	0.05949	0.07662
LSTM+BiLSTM	**0.06949**	**0.04719**	**0.07222**

Table 1: Test loss in bits per token for best-performing models

3.3. Error Analysis

We performed an error analysis by applying the best neural network model to a 10000-token subset of the Irish development set. The total loss in bits per token over this subset was 0.07254, slightly larger than the test loss of 0.06949. The ground truth label was assigned a probability of at least 0.5 for 9833 of the 10000 tokens. In many cases we see that the model has successfully generalized to forms not seen directly in training. For example, no form of the word *ubhal* (a pre-standard spelling of *úll* 'apple') was seen in training, but the model correctly predicts t-prothesis for *an t-ubhal* ('the apple') in the development set. Similarly, the model correctly predicts lenition on the conditional verb *mhaoinfeadh* ('would fund') despite not having seen this word in training, presumably based on other conditional verbs ending in *feadh*.

The remaining 167 tokens (for which the predicted probability of the ground truth label was less than 0.5) account for the great majority (77.24%) of the total loss; we manually examined and classified these tokens into *ad hoc* categories as reported in Table 2.

Classification	Count	Loss (bpt)	% Loss
Error in dev set	61	4.25447	35.78
Right-context needed	30	2.95675	12.23
Non-standard form	23	3.30608	10.48
Possessive *a, ina, ...*	16	2.73679	6.04
Dialect in dative	9	3.18118	3.95
Non-Irish words	7	3.55753	3.43
Others $P < 0.5$	21	1.84388	5.34
All $P \geq 0.5$	9833	0.01651	22.76
TOTAL	10000	0.07254	100.00

Table 2: Breakdown of the total loss on a 10000 token subset of the Irish development set

The greatest part of the loss was contributed by grammatical errors in the development set. In fact, the five tokens with the smallest predicted probability for the ground truth label all correspond to obvious errors. This provides evidence of the usefulness of our model for grammatical error detection and correction, which we report on below in §4.1. Quite a few of the bad predictions stem from our self-imposed restriction that the model only use context to the left. A fluent speaker (and, most likely, a sufficiently strong neural network model) could predict most of these mutations given context on both sides, while failing to do so given only the left context. We can therefore view these particular mutations as conveying information about what comes next in the sentence. Common examples of this type include:

- confusion between imperative (unlenited) and past tense (lenited) verbs at the beginning of a sentence;
- the words *dá* ('if') and *dhá* ('two');
- the preposition *idir* in the sense 'both' (lenition) vs. 'between' (no mutation); and
- the direct vs. indirect relativizing particles, both written as *a*, but distinguished in part by lenition or eclipsis of the following verb.

These and all similar examples are classified as "Right-context needed" in the table.

In our analysis we distinguished true errors from examples that are correct but that do not follow the Official Standard for Irish; the latter are labeled "Non-standard forms" in Table 2. An example would be lenition of the prepositional pronouns *dom*, *duit*, etc., which is common in the spoken language for certain dialects, but is not part of the official orthography.

[5] N.B. TensorFlow uses natural logs, and so the losses shown in training are smaller by a factor of $\ln(2)$ than those reported in Table 1.

The next two rows in Table 2 correspond to the mutations noted in §2.3 as (sometimes) carrying information; namely, the possessive *a* along with its various forms fused with prepositions (*ina*, *faoina*, *dá*, etc.), and the choice of lenition or eclipsis on nouns in the dative case based on dialect (both choices being permitted in the Official Standard).
The final class consists of non-Irish (mostly English) words which do not follow Irish spelling conventions and therefore confuse the model.

3.4. Model Introspection

The two best neural network models learn 200-dimensional embeddings for all tokens in the vocabulary. By visualizing these embeddings we can verify that the model is learning various linguistic properties that we know *a priori* to be important in the mutation system.
For example, Figure 2 is a two-dimensional projection of the embedding space (using tSNE) in which we have plotted 7242 Irish nouns (nominative forms only) according to their gender: blue dots are feminine and red dots are masculine. At the most basic level, it is encouraging that words of each gender tend to cluster together since gender plays a key role in determining the correct mutation on nouns and on the adjectives that modify them.
Interestingly, the visualization reveals additional regularities that the model learns at the character level. For example, there are subclusters visible within each gender; these are made up of words whose initial letters behave similarly with respect to mutations. Words with initial vowels form a tight subcluster within each gender, for example, as do nouns with initial *l*, *n*, *r*, *sc*, *sm*, or *sp* (none of which mutate). Similarly, there are subclusters of nouns with initial *b*, *c*, *f*, *g*, *p* (consonants that admit lenition and eclipsis in the same contexts), and another consisting of nouns with initial *d* or *t* (which also admit lenition and eclipsis, but not following certain words like the definite article *an*).

4. Applications

4.1. Grammar Checking

In this section we evaluate the ability of the best-performing model to detect grammatical errors in Irish.
There is an active community of Irish speakers on Twitter, with more than 3.3 million Irish language tweets posted through the end of October 2019.[6] The informal nature of social media combined with the presence of many language learners makes Twitter an excellent resource for creating a corpus of grammatical errors. To this end, we examined a random sample of about 80,000 Irish language tweets for mutation errors, and from these selected a gold-standard corpus of 895 tweets containing 1029 mutation errors.[7] The tweets were lowercased and tokenized in the same way as our training corpora in §3, yielding 14406 tokens. Tokens beginning with @ (representing Twitter usernames) were preserved, but were all converted to a single token (*@twitter*) as a way of partially anonymizing the corpus. We did *not* correct any of the errors, but simply flagged all tokens for which we believe the "ground truth" mutation label to be an error.
Then, for various probability cutoff values $0 < C < 1$, we apply the LSTM+BiLSTM model to the corpus and instruct the model to report an error any time it assigns a probability less than C to the ground truth label. The precision, recall, and F-scores for detecting mutation errors are given in Table 3 at cutoff increments of 0.1, and plotted as a precision-recall curve at increments of 0.05 in Figure 3.

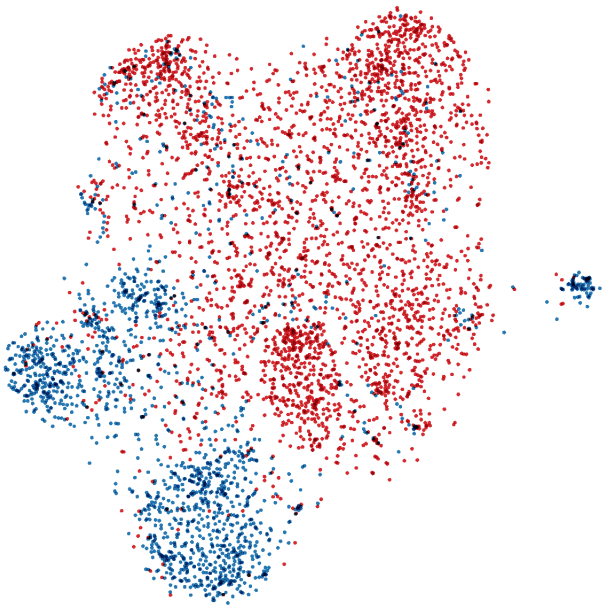

Figure 2: 2D projection of word vectors for 7242 Irish nouns in their nominative form; blue are feminine and red are masculine.

$P < C$	Precision	Recall	F-score
0.1	0.93	0.66	0.771
0.2	0.91	0.80	0.853
0.3	0.90	0.87	0.885
0.4	0.87	0.90	0.887
0.5	0.85	0.93	0.889
0.6	0.82	0.94	0.880
0.7	0.80	0.96	0.868
0.8	0.75	0.96	0.844
0.9	0.66	0.97	0.781

Table 3: Precision, recall, and F-scores for mutation error detection at given probability cutoffs

The lowest cutoff used in our experiments was probability C=0.05, and in this case there were 38 false positives reported. For many of these, the model was led astray by informal language found in tweets that is not well-represented in our training corpora. For example, 10 of the 38 false positives are exclamations written according to English con-

[6] See `http://indigenoustweets.com/ga/`.
[7] Available from `https://github.com/kscanne/gramadoir/blob/master/ga/corpas.json`. Mutations that are acceptable in at least one dialect were *not* included in the error corpus, despite not conforming to the Official Standard in some cases.

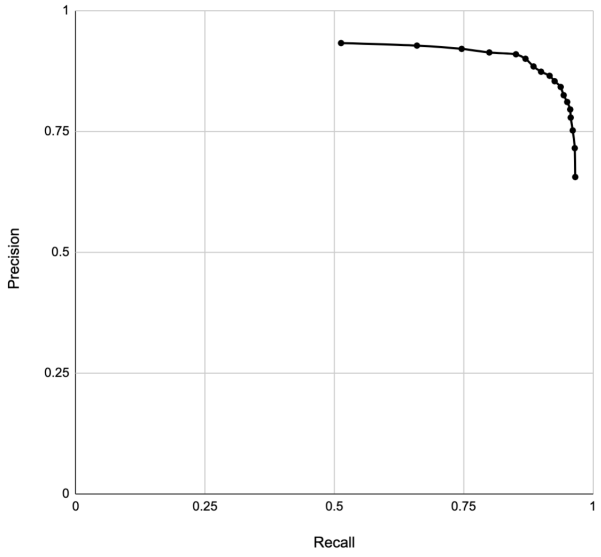

Figure 3: Precision/Recall curve for error detection

ventions (*ah*, *hi*, *oh*, *ha*), and several others are caused by misspellings that distorted a word enough to confuse the model, e.g. *bhanada* for *bhanda* ('band', lenited) or *Ghàidlig* for *Ghàidhlig* (the autonym for Scottish Gaelic, lenited).

We remind the reader again that our neural network makes its predictions based only on the left context, whereas a typical batch-style grammar checking application would be able to make use of both left and right context. Therefore it is likely that a model tailored to this particular application could improve upon the results in this section.

4.2. Language Modeling

Given a vocabulary $\mathcal{V}$ of tokens, a language model assigns a probability $P(v_1 \dots v_N)$ to any sequence $v_1, \dots, v_N \in \mathcal{V}$. These are usually computed as a product of token probabilities conditioned on their histories:

$$P(v_1 \dots v_N) = \prod_{i=1}^{N} P(v_i | v_1 \dots v_{i-1}) \tag{2}$$

Given a test corpus $\mathbb{T} = w_1 \dots w_N$, a language model can be evaluated using the per-token cross entropy $h(P, \mathbb{T})$ (cf. Equation 1):

$$h(P, \mathbb{T}) = -\frac{1}{N} \sum_{i=1}^{N} \log_2 P(w_i | w_1 \dots w_{i-1}), \tag{3}$$

or, more commonly, the perplexity:

$$PPL = 2^{h(P,\mathbb{T})} = \left[\prod_{i=1}^{N} P(w_i | w_1 \dots w_{i-1}) \right]^{-\frac{1}{N}} \tag{4}$$

This paper arose, more or less, from the following simple observation. A 5-gram language model with modified Kneser-Ney smoothing (Chen and Goodman, 1999) trained on *demutated* Irish text gives a better than 6% improvement in perplexity over the same model trained on raw text, "for free". That we see some improvement should come as no surprise since demutation allows the model to learn generalizations across words in the training set that would otherwise differ; e.g. an important collocation like *bád seoil* can be learned or strengthened even from mutated training examples like *mo bhád seoil* ('my sailboat', lenited) or *ár mbád seoil* ('our sailboat', eclipsed).

The idea then is that a sufficiently accurate model for predicting mutations allows us to realize most of this 6% improvement in a language model on the original corpus, without demutation. We achieve this by means of a *factored language model*, following (Bilmes and Kirchhoff, 2003). In our setup, we would like to model each surface token w as a pair of features, its unmutated form $f_1 = \sigma(w)$ and its mutation $f_2 = \mu(w)$. Given a sequence of tokens $w_1 \dots w_k$, we factor the probability $P(w_k | w_1 \dots w_{k-1})$ as follows:

$$\begin{aligned} P(w_k | w_1 \dots w_{k-1}) &= P(\sigma(w_k) | w_1 \dots w_{k-1}) \cdot \\ &\quad P(\mu(w_k) | w_1 \dots w_{k-1} \sigma(w_k)) \\ &\approx P(\sigma(w_k) | \sigma(w_1) \dots \sigma(w_{k-1})) \cdot \\ &\quad P(\mu(w_k) | \sigma(w_1) \dots \sigma(w_k)) \end{aligned}$$

The first factor here is simply a language model trained on demutated text, while the second term is a mutation model of exactly the type developed in §3.

We experiment only on Irish, making use of the clean Tuairisc.ie corpus described in §3.2 upon which the mutation model works the best. We hold out development and test sets containing 750k tokens each, and use the remaining 6.2M tokens for training. The n-gram models were trained and evaluated with the MITLM toolkit (Hsu and Glass, 2008) using modified Kneser-Ney smoothing. We used the full training vocabulary (95905 tokens in the base corpus and 75902 in the demutated corpus) and tuned parameters to minimize perplexity on the development set. The results are presented in Table 4.

Model	PPL (dev)	PPL (test)
KN 5-grams (raw)	75.04	74.10
KN 5-grams (demutated)	70.28	69.53
Factored LM	72.81	72.03

Table 4: Perplexities of Irish language models

The drop in perplexity from 74.10 for the baseline model to 69.53 for the demutated model is the roughly 6% improvement noted above, occurring presumably because the latter model is able to learn generalizations across words differing only in mutations. The perplexity of 72.03 for the factored language model shows that an accurate model for mutation prediction allows a good chunk of this improvement to be realized in a language model on the original corpus. With further improvements to the mutation prediction model, we would expect the perplexity of the factored language model to move even closer to the perplexity of the demutated model.

There is nothing special about the 5-gram language model as far as the formal setup; it is possible to similarly factor out initial mutations with state-of-the-art neural language models. It remains to be seen whether a comparable perplexity improvement is achievable in that case, since a character-aware neural model may be able to learn to predict initial mutations effectively despite not being explicitly trained to do so. We will return to this question in forthcoming work in which we evaluate a wide variety of neural language models for the Celtic languages.

5. Acknowledgements

This work was completed while visiting Acadamh na hOllscolaíochta Gaeilge in Carna, Co. Galway, Ireland as a Fulbright Senior Scholar. I am grateful to the Fulbright Commission in Ireland and the staff of the Acadamh, especially Séamas Ó Concheanainn, for their hospitality and for making the visit possible. I would also like to thank Mícheál Ó Meachair, Brian Ó Conchubhair, Cathal Convery, Teresa Lynn, and Michael Goldwasser for their help and encouragement during the Fulbright application process.

6. Bibliographical References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, pages 265–283.

Bauer, M. (2011). *Blas na Gàidhlig: The Practical Guide to Gaelic Pronunciation*. Akerbeltz, Glasgow.

Bilmes, J. A. and Kirchhoff, K. (2003). Factored language models and generalized parallel backoff. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003–short papers-Volume 2*, pages 4–6. Association for Computational Linguistics.

Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394.

Evas, J. (2013). *The Welsh Language in the Digital Age / Y Gymraeg yn yr Oes Ddigidol*. Springer.

Graves, A. and Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5–6):602–610.

Hsu, B.-J. and Glass, J. (2008). Iterative language model estimation: efficient data structure & algorithms. In *Ninth Annual Conference of the International Speech Communication Association*.

Judge, J., Ní Chasaide, A., Ní Dhubhda, R., Uí Dhonnchadha, E., and Scannell, K. P. (2012). *The Irish Language in the Digital Age / An Ghaeilge sa Ré Dhigiteach*. Springer.

Scannell, K. P. (2007). The Crúbadán Project: Corpus building for under-resourced languages. In *Building and Exploring Web Corpora: Proceedings of the 3rd Web as Corpus Workshop*, volume 4, pages 5–15.

Tithe an Oireachtais. (2016). *Gramadach na Gaeilge: An Caighdeán Oifigiúil*. Baile Átha Cliath.