IMPROVING SPELLCHECKING FOR CHILDREN: CORRECTION AND DESIGN

by

Brody Downs

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Computer Science

Boise State University

August 2020

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Brody Downs

Thesis Title: Improving Spellchecking for Children: Correction and Design

Date of Final Oral Examination: 6 July 2020

The following individuals read and discussed the thesis submitted by student Brody Downs, and they evaluated the presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Jerry Fails, Ph.D.	Chair, Supervisory Committee
Casey Kennington, Ph.D.	Member, Supervisory Committee
Maria Soledad Pera, Ph.D.	Member, Supervisory Committee
Katherine Wright, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Jerry Fails, Ph.D., Chair of the Supervisory Committee. The thesis was approved by the Graduate College.

ACKNOWLEDGMENTS

I am sincerely grateful to my advisor, Dr. Jerry Fails, for his support and guidance in the completion of my thesis and Master's degree. I am also extremely appreciative for the opportunities he provided through a graduate assistantship that allowed me to pursue research, complete my Master's degree, as well as attend and present at conferences and workshops. I must also thank my supervisory committee members Dr. Casey Kennington, Dr. Sole Pera, and Dr. Katherine Landau Wright each for their roles in the completion of my thesis, graduate experience, and unique feedback stemming from their respective disciplines.

I would also like to thank the current and past members of the CAST Team who I had the great pleasure of working with and learning from: Tyler French, Mikey Krentz, Aprajita Shukla, Oghenemaro Anuyah, and Garrett Allen.

This research has been partially supported by the National Science Foundation (Award #1763649).

ABSTRACT

Children commonly use software applications such as search engines and word processors in the classroom environment. However, a major barrier to using these programs successfully is the ability of children to type and spell effectively. While many programs make use of spellcheckers to provide spelling corrections to their users, they are designed for more traditional users (i.e., adults) and have proven inadequate for children. The aim of this work is twofold: first, to address the types of spelling errors children make by researching, developing, and evaluating algorithms to generate and rank candidate spelling suggestions; and second, to evaluate the impact interactive elements have on children's spellchecking behaviors seeking to improve their user experience.

Motivated by children's phonological strategies to spell, a phonetic encoding strategy is used to map words and misspellings to phonetic keys to effectively and efficiently provide spelling correction candidates. Machine learning methods, including Learning to Rank, are used to rank candidates effectively and reveal the importance of phonetic features. Experimental results show this method is able to more accurately provide and rank spelling corrections when handling misspellings generated by children in both essay writing and web search settings when compared to state-of-the-art baselines. The design of an interactive spellchecker reveals children's propensity towards visual and audio cues. A study on visual and audio cues show an influence on children's selection habits and a positive impact on assisting children in selecting correct spelling suggestions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

As children are introduced to the web at increasingly younger ages [46], search engines have become a valuable resource for information discovery [4]. However, popular search engines, including Google or Bing, are not built with children in mind which impairs children's ability to interact with them effectively [30, 28]. Particularly challenging is the initial point in the search process when children must express their need to the search engine via a query. At that crucial point, children frequently struggle with the process of spelling and typing which is an issue given that the use of correct terminology is pivotal in formulating queries that a search engine can effectively process [34, 17]. In fact, reports indicate that between 25% and 40% of queries made by children contain at least one spelling error [29]. Consequently, the presence of spelling errors in a query can cause search engines to not only retrieve resources irrelevant to the user but may also result in empty search engine result pages [21, 55]. Given children's difficulties with spelling and the importance of correctly spelled queries, it becomes imperative that the spelling needs of children be addressed.

To assist in correcting spelling errors, it is a common approach of search tools to make use of spellcheckers [10] to help capture the user's intent [46]. While research has allocated efforts to address and improve spellchecking, either in query formulation for search engines [1, 45] or in essay writing for word processing tools [43, 41],

there has been a lack of research that targets young children in particular. As research has shown, the use of traditional or mainstream spellcheckers are inadequate when correcting spelling errors formed by children in search queries [14] and typical interfaces do not support children interacting with spellcheckers [15]. This emphasizes the importance of research in child-oriented spelling correction that could be applied particularly to search and could have broader application to other contexts to assist children in their spelling needs. As such, this research investigates how to improve spelling correction and spellchecking interfaces for children.

1.1 Spelling Correction

To address the issue of spellchecking, it is common to split the issue into two sub-problems: spelling error *detection* and spelling error *correction*. In spelling error detection, spellcheckers are tasked with finding and correctly identifying words that have been spelled incorrectly. Many contemporary spellcheckers approach this by simply searching for typed words, in isolation, in a set of known correct words (i.e., a lexicon) [13]. This approach is often not sufficient, especially when dealing with homonyms, grammar mistakes, or real-word errors, which require grammatical analysis or sequence modeling [25, 57]. While important, significant work has been done on detecting these errors [25, 57, 56, 45], therefore this work focuses instead on the spelling correction aspect.

Once a spelling error has been detected, we must then make an effort to correct the spelling error by identifying the word, or possible words, the user intended to spell (i.e., spelling suggestions). Common to many spellchecking strategies is to first generate a list of suitable candidates, then rank those candidates in an order such that

the most likely suggestions appear at the top of the list of suggestions [25, 45]. What most strategies often overlook or ignore, is that the spelling behavior of children is different from adults.[31]. As a result, the common techniques used for spell correction are often inadequate when correcting the unique spelling errors made by children [38]. Addressing children's spelling behavior and errors is a key part of our research for effective spelling correction. This research looks to further investigate and analyze the effect of candidate generation on children's spelling errors by addressing the phonetic structure of words that aligns with how children spell and improve the ranking of spelling suggestion candidates through various machine learning methods. These are covered in Chapter 4.

1.2 User Interaction

Once a list of spelling suggestions has been generated based on a detected spelling error, the spellchecking system may correct spelling errors either interactively or automatically [26]. In an interactive system, spelling suggestions will be presented to the user from which they then choose the suggestion that correctly matches their intended word. In an automatic spellchecking system, the highest ranked word determined by the correction algorithm may be chosen and used to replace the spelling error without input from the user. Many modern spellcheckers often aim to correct spelling errors quickly and efficiently, through automatic correction, rather than helping users develop spelling skills which conflicts with research-based best practice for spelling correction [37]. Previous research showed that children can struggle with identifying and addressing misspelled terms as well as selecting their intended word from a list of spelling suggestions [14]. When interacting with a list of

spelling suggestions, children clicked on their intended word just 67% of the time [14]. This highlights that there are not only issues with providing children with suitable spelling suggestions, but also assisting them in the spellchecking process. Following research and guidelines in child computer interaction, we use audio and visual cues to assist with the selection of spelling suggestions. We further discover other problems related to spellchecking interfaces and propose solutions going forward. These are covered in Section 6.

1.3 Research Overview – Research Questions and Plan

To summarize, children's difficulties with spelling when compared with adults limits their ability to use search engines effectively. Although search engines may make use of spellcheckers, they don't consider a child audience. Due to how children's spelling behaviors and errors differ from adults the effectiveness of state-of-the-art spellcheckers is limited. Furthermore, interfaces are not supportive of children when making spelling corrections. Research has documented these problems, but little has been done to address them.

The goal of this research is to address the issues children face with spellcheckers both in terms of correcting their spelling and when interacting with a user interface. Specifically we answer the following questions:

- Do approaches that align with children's spelling behaviors improve spelling candidate generation?

- What models work best for re-ranking spelling suggestion candidates?

- What features are most important for ranking spelling suggestion candidates for children?
- Are audio and visual cues effective in assisting children in making spelling suggestion selections?
- What other problems do children face when using spellchecking interfaces and what steps can we take to resolve those going forward?

To accomplish this we perform research, development, and evaluations on several tasks. We address spelling correction by using phonetic spellchecking strategies to effectively and efficiently generate spelling candidates. We then research the relative advantage of different machine learning model with features designed towards a child user to improve ranking. These methods are extensively evaluated against state-of-the-art models both for generating and ranking candidates. In order to address children's selection behavior while interacting with a list of spelling suggestions, we investigate the impact of audio and visual cues. Through the use of participatory design involving children as design partners, we discover other issues related to spellchecking interfaces and propose steps going forward to solve them. The outcomes of our studies, evaluations, and analysis advance our understanding of how to better appeal to and support children's spellchecking needs.

CHAPTER 2

RELATED WORK

In this chapter, we review related work that analyzes the spelling strategies of children compared to adults and the common types of spelling errors made by users. We then review prior work on traditional and state-of-the-art methods for correcting spelling errors. Lastly, we cover prior work addressing children's interactions with computers and how that can be used to guide our design for a more effective spellchecking tool.

2.1 Spelling Strategies and Errors

To approach spelling correction it is necessary to understand the spelling process and the types of spelling errors being made, which will then allow us to take steps to undo those errors [13]. Greenberg et al. reported that when compared to adults, children tend to use more phonological strategies (i.e., spelling by using sounds) and less orthographic processes (i.e., memorizing letter sequences associated with individual words) when spelling [31]. As a result, the spelling errors made by children often differ from adults, which spellcheckers often don't consider. Deorowicz et al. [13] reported on three different types of spelling errors that occur in the spelling and typing process: vocabulary incompetence, misspellings, and mistypings. **Vocabulary incompetence** errors often occur when attempting to create a word using known prefixes or suffixes along with known root words. For example, one may want

to want to write the negative form of *perfect* resulting in the incorrect forms of *unperfect*, *inperfect*, or *aperfect*. **Misspellings** occur when a user is unsure of the spelling or pronunciation and try to form the phonetically (i.e., *grammer* instead of *grammar*). **Mistypings** occur when the user knows the spelling but makes an error while pressing the keys (i.e, *spwlling* instead of *spelling*). Although, it has been noted that misspellings and vocabulary incompetence are common types of errors formed by children [13], these types of errors may not encompass all spellings errors made by children.

2.2 Spelling Correction

Little research has been done on correcting errors made by children or the effectiveness of spellchecking strategies on children's errors, but a look at the traditional methods for spellchecking informs us of some possible solutions. An overview of traditional methods of spellchecking are covered by Deorowicz et al. [13], in which they categorize spellchecking strategies into several techniques: edit distance, similarity keys, rule-based, probabilistic, and phonetic similarity.

Edit distance methods look to correct spelling errors by addressing the amount of basic editing operations needed to go from a correctly spelled word to the spelling error. These operations, introduced by Damerau [11] and Levenshtein [44] are defined as follows: *insertion* of a letter, *deletion* of a letter, *substitution* of one letter for another, and *transposition* of two adjacent letters. As noted in previous research [14], spellcheckers using simple edit distance methods are ineffective at correcting children's spelling errors.

Similarity key methods use techniques to assign a key to each word in a given

dictionary. The key generated for the spelling error is then used to find words that have a matching key, which are used as spelling suggestions. This technique can be especially efficient when used in conjunction with other methods, such as edit distance, by limiting the number of words to process in order to compare to the spelling error. Techniques of this kind, such as *SoundEx*, have been used when correcting spelling errors made in search queries [10].

Rule-based techniques use known and common spelling errors to transform the spelling error into possible real words. Yannakoudakis and Fawthrop [57] created a system using rules from experiments containing over 1000 spelling errors. The rules that are created are applied to spelling errors in order to produce a list spelling suggestions. However, applying these rules to generate a list of words can be inefficient, which is a problematic for real-time spellchecking [13]. Additionally, while these can be used to correct many different types of spelling errors, they require large amounts of data to build a sufficient rule base which is unavailable for children's spelling errors.

Probabilistic techniques try to determine the probability of a word w given a spelling error e typically using the derived Bayes Thereom $P(w)*P(e \mid w)$ where $P(w)$ represents a language model that determines the probability that w appears in the text and $P(e \mid w)$ represents an error model determining the probability that e is typed when the user meant w [7]. Some techniques use the probabilities of making certain types of errors. For example, Church and Gale [9], obtained the probabilities of the basic editing operations on certain letters. Inserting one key over another could be based on the proximity of those keys on a keyboard. For example, incorrectly substituting an "a" for an "s" may be much more likely than substituting an "a" for a "p" due to their proximity on a QWERTY keyboard. Building a knowledge base for the probabilities on these types of errors would require a large corpus of data on

common spelling errors, which is largely unavaiable for children. Other probabilistic techniques make use of language models and word context [10]. Given the phrase "golf curse" it is likely the user meant "golf course" despite the initial phrase being spelled correctly. Brill and Moore [7] use an improved noisy channel that takes advantage of common string substitutions rather than single character edit operations to improve spellchecking.

Phonetic similarity methods look to correct spelling errors made when a user knows the pronunciation of a word, but not the correct spelling, as such they produce spelling errors with incorrect graphemes. The previously mentioned SoundEx [10] as well as the PHONIX [24] method use similarity key techniques to produce words with similar phonetic pronunciation. Other algorithms to produce phonetic keys to index words, such as Metaphone [50], have been used in the Aspell spellchecker [3]. While effective at correcting misspellings, a common error made by children, they are not tuned for children and often fail to pick up other types of errors including mistypings and incompetence errors [13].

More recently, **machine learning** methods have been applied to correct spelling errors. De Amorim et al. [12] and Pande [49] both applied machine learning techniques for spelling *candidate generation* in an effort to reduce the number of distance metrics that must be calculated when comparing a spelling error to words from a dictionary corpus. As more complex models will require more computing power, a smaller search space will be necessary for efficient spellchecking. De Amorim et al.[12] used unsupervised clustering of words. Their methods show an improvement over state-of-the-art methods when correcting errors from a dataset that include a combination of both adult and child spelling errors but their reported results make no distinction between the two. While their method uses edit distances as a base for the

word clustering, any distance measure valid for strings could be used, and considering that edit distance methods are effective for children [14, 13], other distance metrics could be employed. Pande [49] used character string embeddings who showed a performance increase over the work of De Amorim et al. with impressive efficiency. While successful on a mixture of spelling errors, they have not showcased success for children's spelling errors. This work could be improved by using knowledge we know of children's spelling from previous research, incorporating phonetics or ignoring vowels in embedding generation.

Other machine learning methods have been focused on ranking and candidate selection. The use of classifiers in spellchecking for spelling correction and candidate ranking has shown to outperform state-of-the-art spellchecking systems as described in the work by Fomin and Bondarenko [22] as well as Huang et al. [36]. Huang et al. [36] explore neural network classifiers using keyboard layout and edit distance as word features to correct spelling errors made in the automotive domain showing that their method outperforms other state-of-art methods. While their methods look to correct spelling errors made by adults rather than children, they work in a unique domain that requires a knowledge base different than a general-purpose spellchecker, which could be liked to children's spelling errors [36]. Fomin and Bondarenko [22] employ a similar method using logistic regression and a large feature set. While they make use of features like edit distance, n-gram probabilities, and keyboard distance, they do not consider the phonological structure of words. Based on our previous work, we believe the development of a feature set representing phonetic features of words that more closely represent the similarity of children's misspellings and their intended word could improve the performance of these models for children's spelling errors.

Li et al. [45] take a different approach, leveraging Hidden Markov Models to that looks to model several types of spelling errors that includes the merging and splitting of words which children tend to have an issues with [14]. However, as with the previous machine learning methods, they limit themselves to edit distance metrics for identifying spelling correction candidates. Ganjisaffar [25] utilize similar methods discussed previously to generate a large set of candidates and then focus on ranking those candidates appropriately use various language model features. While effective for correcting search queries, there is a reliance on a large amount of textual data that is adult oriented. Whitelaw et al. [56] use a similar n-gram language model approach that makes use of web data rather than manually annotated data, which allows it to be easily implemented for other languages as well as handle unique terms.

In each of the papers described, the focus and data used, is on adult spelling errors. Although they may be effective for adult spelling, they do not conform to the types of errors children make which are unique from those of traditional users, particularly in web search settings [29]. The lack of spelling data for children creates an issue for addressing this task and training several of the mentioned models. This makes collecting data on spelling errors formed by children essential our research.

2.3 User Interaction

In addition to the algorithmic perspective, we look to improve the experience children have when interacting with a spellchecker. Although little research has been done exploring children's experience with spellcheckers, we may look to the work done in the field of child-computer interaction. Druin et al. [17] advocates for interactive spelling assistance in web search settings. To assist in the design of an interactive spellchecking

tool, we make use of the methods and techniques used in child-computer interaction research that involves including children in the design process as participatory design partners [20, 32, 35]. To get an idea of what may make a more effective tool for children, we look to interaction guidelines which emphasize the importance of speaking the user's language, which for children may not be text, but sounds and images [35].

While many researchers have shown children's preference for visual interfaces [39] and support the use of multiple types of input (i.e., images) [17, 30], there has yet to be research that employs these methods in the context of a spellchecking interface. Additionally, children use a memorization strategy to link concepts through visual images to store information in long-term memory, which can be leveraged to aid in children's learning [38].

Sluis et al. [53] used sounds/phonemes in an application to help children match words to sounds in an effort to enhance a child's consciousness of phonological units and enforce their reading skills. Michaelis and Mutlu [47] explore the use of social robots to assist in reading science textbooks showing that the use of expressive synthesized speech assisted children by boosting their interest and understanding of science from textbooks. As such, the use of sound cues or synthesized speech show promise in improving children's overall engagement when interacting with text-based technology such as a spellchecker.

Another important aspect that impacts the selection of spelling suggestions from a list is the order they are displayed. It has been shown that the ranking of vertically positioned options influences children's choice [29, 18], who show a bias of options that are oriented higher. Similarly, our previous research has shown children tend to favor the higher ranked spelling suggestions, regardless if they are correct [14].

This emphasizes the importance of not only finding correct spelling suggestions but ranking them appropriately.

The studies presented suggest that search engine interfaces need to be intuitive and engaging for children. Given children's preferences and attentiveness towards audio and visual cues, this seems like a promising route, but has yet to be explored in a spellchecking interface. In the studies below, we highlight how these cues can be beneficial. These findings also align with Dual Coding Theory [51], an established theory of learning that posits that providing information in multiple modalities aids readers comprehension. We take inspiration from these works to assist children in a spellchecking interface.

CHAPTER 3

DATA COLLECTION

Essential to this study is a collection of child-made spelling errors. For this, we use spelling errors collected from children in two different contexts: hand-written essays and typed search queries, which we have summarized in Table 3.1. The following sections describe each of the datasets and the collection process for each.

3.1 Children's Spelling Errors in Hand-Written Essays

The hand-written essay spelling error dataset was built based on writing samples from 82 children collected at a university-based literacy clinic. During each of the Fall and Spring semesters, children in grades K through 8 take part in this clinic to receive one-on-one and small group tutoring from undergraduate students. The students tutoring in these sessions are pursuing elementary education licensure. The children have diverse backgrounds: some are English language learners and some have learning disabilities. English language learners speak a variety of native languages including

Table 3.1: Summary of children's spelling error datasets.

Dataset name	# of misspellings	Source	Attributes
Essay$_{\text{MSP}}$	1651	hand-written essays	misspelled word, correct spelling, grade, spelling level, native language, words before
Query$_{\text{MSP}}$	134	typed search queries	misspelled word, correct spelling, clicked word, words before, sessionID
Childrens$_{\text{MSP}}$	1785	combination of ESSAY$_{\text{MSP}}$ and QUERY$_{\text{MSP}}$	misspelled word, correct spelling, words before

Italian, Spanish, Korean, Japanese, and Mandarin. Outside of the native English speakers, the group of native Korean speakers is the most diverse. The majority of the data from the other non-English native speakers come from few students (42 English, 30 Korean, 2 Italian, 2 Spanish, 2 Japanese, 1 Mandrin).

All children independently complete a hand-written writing sample at the beginning of each of the semesters. If a child's hand-writing is difficult to decipher, tutors will ask the child what they wrote and transcribe their writing, but no corrections are made to the child's original writing. This process is periodically repeated throughout the semester to measure the students' progress. Each writing sample is transcribed digitally and annotated for potential spelling errors. The spelling errors examined for this study were collected over three years (i.e., seven semesters).

For each misspelled word that was recorded (i.e., digitized), the dataset includes their intended word, grade, spelling development level, and the student's native language. For a portion of the entries (485 out of 1651) up to three words proceeding the misspelling was also recorded. Spelling development levels are recorded as one of the following: Emergent, Letter Name-Alphabetic, Within Word Pattern, Syllables and Affixes, and Derivational Relations based on known progression stages for spelling development and behavior [5]. The resulting dataset consists of 1,651 misspelled words along with their corresponding spelling corrections after removing duplicate entries. This dataset is refered to as $\mathbf{Essay_{MSP}}$. Information regarding this dataset is compiled in Table 3.1. Examples of this dataset can be seen in Table 3.1.

Table 3.2: Sample of instances in EssayMSP

Target	Spelling	Grade	Level	Language	Words Before
always	olwes	4	Letter Name Alphabetic	Spanish	like she
differences	diffrences	3	Syllables and Affixes	Korean	similarities and
professor	pfes	3	Within Word Pattern	English	was it

3.2 Children's Spelling Errors in Typed Search Queries

The data in this section was collected as part of experiments in previous research conducted on children's spelling [14, 15]. The child participants in these sessions used a custom search tool on a desktop computer for entering search queries. To elicit children's search queries, they were given various open-ended and fact-based prompts (examples can be seen in Table 3.3). The custom search tool provides spellchecking utility that will mark spelling errors and provide spelling suggestions.

Table 3.3: Sample tasks prompts given to child participants

Search Task	Type
Fact-based	Who was the first computer programmer? Who was the scientist that invented robots? How far away is the Sun? How tall is a tyrannosaurus rex?
Open-ended	Find me a cool fact about space. Find me a difference between Earth and Mars. Find me an interesting fact about Albert Einstein. Find me a fact about your favorite dinosaur.

During query formulation, if a spelling error is identified, it will be marked as such by being underlined and colored in red. Hovering over any spelling errors will provide a list of up to 5 spelling suggestions. Dependent on the settings, hovering over a suggestion may also display a relevant image and/or read the word aloud using a speech synthesizer. Clicking on a spelling suggestion replaces the spelling error with the suggested spelling. An example of the spellchecking interface can be seen in Figure 3.1. All user interactions with the interface were automatically recorded. Ad-ditionally, facilitators (made up of graduate and undergraduate researchers) observed and recorded notes based on interactions children made with the search interface with a focus on words children misspelled. As the spelling suggestion that matches the

child's intended word may not always appear on the list of suggestions and because the suggestion children click does not always match the word they intended to spell, their intended word had to be determined. The intended word for each misspelling was collectively agreed upon by four facilitators after the experiments had been completed based on query logs, search prompts given, and notes taken during sessions. They were then validated by an expert in children's literacy.

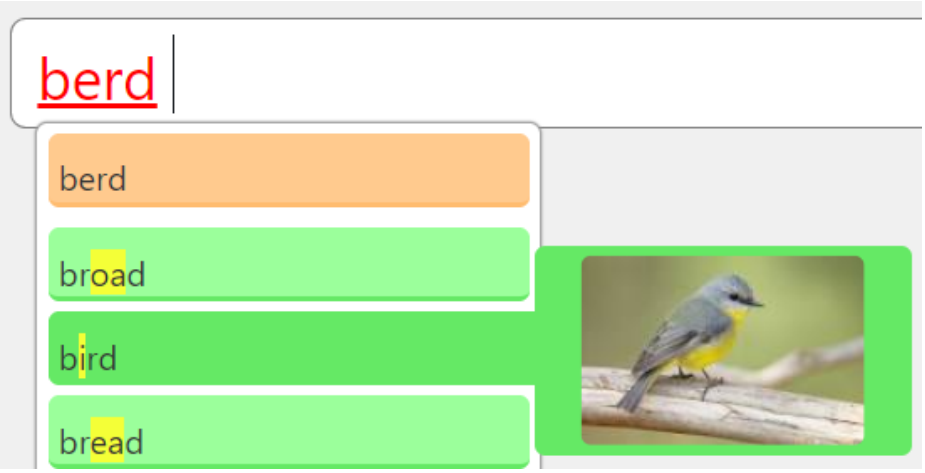

Figure 3.1: Example of the search interface with spelling error and suggestions

For each spelling error recorded, the dataset created includes the spelling suggestion that was clicked on, the agreed upon intended word, up to three words before the spelling error, and the users session ID. The resulting dataset includes 134 entries along with the mentioned attributes after removing duplicates. This dataset is referred to as $\mathbf{Query_{MSP}}$. Information regarding this dataset is compiled in Table 3.1. Examples of this dataset can be seen in Table 3.2.

Table 3.4: Sample of instances in $\mathbf{Query_{MSP}}$

Target	**Spelling**	**Clicked**	**Words Before**
specific	pisific	pacific	was the
einstein	enistein	einstein	did albert
invented robots	evendedrobots	n/a	who

CHAPTER 4

SPELLING CORRECTION METHOD

In this chapter, we cover in detail our English, child-oriented spellchecking method. We first discuss the method for candidate generation using a phonetic encoding algorithm, which includes details on the creation of our dictionary. We term the candidate generation method as **KidSpell**. This method simply returns generated candidates in order of term frequency. We improve on the ranking of candidates using the lambdaMART Learning to Rank model. This method with improved ranking is termed **KidSpell**λ. At a high level, given a spelling error from a child, our model will apply a phonetic similarity approach to generate suitable candidates then re-rank those candidates based on a number of features. The architecture of the model is illustrated in Figure 4.1.

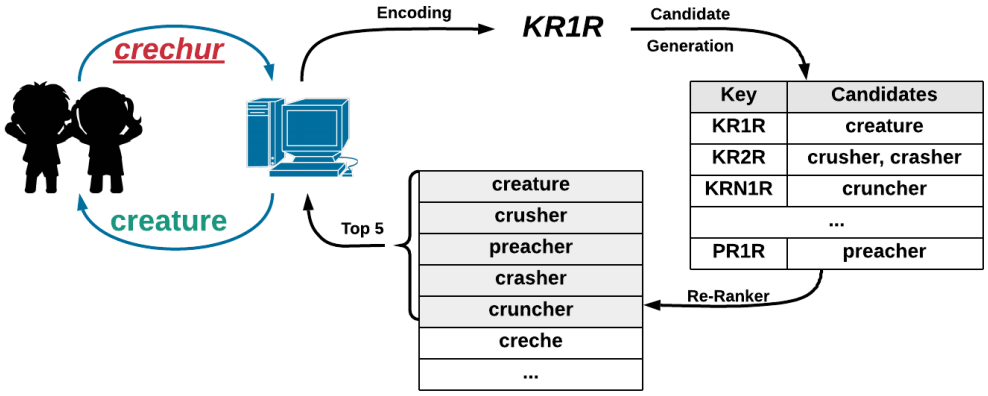

Figure 4.1: KidSpellλ architecture using the spelling error *crechur* for the word *creature*

4.1 Candidate Generation

The goal of the candidate generation is to reduce the search space such that a spelling correction method can efficiently rank the given candidates rather than observing and processing every word in the dictionary. As children have a tendency to use phonological strategies over orthographic ones [31], spelling correction methods that use a phonological approach show promise. As such, we take inspiration from Deorowicz and Ciura who showed that phonetic similarity strategies are effective in correcting spelling errors made by those who know the pronunciation, but not the spelling, which is a common error made by children [13]. At a high level, given a misspelled term written by a child, our candidate generation model applies a phonetic similarity approach in order to identify relevant spelling candidates.

4.1.1 Dictionary Creation

Critical to most spellchecking methods is a list of valid words, known as a *dictionary*. We create child-friendly dictionary that is lacking in typical spellcheckers. Considering

that children typically acquire around 60 thousand words in their first 18 years life [6], a limited lexicon of child-known words could assist in providing more reasonable spelling suggestions. To achieve this we derive our dictionary from age of acquisition research which lists and identifies the typical age words are learned [40].

A child-friendly spellchecker should also refrain from producing any sexually explicit, hate-based, or other inappropriate words. An investigation into the extent at which spellcheckers produce spelling suggestions that include sexually explicit or hate-based words showed many popular spellcheckers have a tendency to produce such words in up to 5% of their suggestions [14] (See Table 4.1). Such words were removed from the dictionary's pool of possible suggestions. Words were determined to be sexually explicit in nature if they exist in a dictionary of sexually explicit words created based on Google's bad words list.[1] Hate-based words were identified as those that exist among the list of hate-speech and offensive language lexicons which we compiled from HateBase,[2] a repository of hate-speech language.

Table 4.1: Sexually explicit and hate-based word rate in top 5 suggestions on various spellcheckers.

Enchant	SimSpell	Bing	Hunspell	Gingerit	Aspell
Hate-based words					
0.0156	0.0264	0.00293	0.0156	0.0039	0.0234
Sexually explicit words					
0.045	0.0489	0.00097	0.04207	0.0078	0.0469

There are limitations to removing such words. Some of the sexually explicit terms included among the Google's bad word list are ambiguous in nature and may not necessarily be inappropriate when considered in certain educational contexts, e.g., *rectum* and *screw*. Furthermore, some misconstrued hate-based words provided by

[1] https://code.google.com/archive/p/badwordslist/
[2] https://www.hatebase.org/

the HateBase repository include *bumblebee* and *slave*, which may pertain to some classroom terminology.

In the classroom context, we perceived that preventing children's exposure to a potentially false positive word (i.e., a word that may be relevant to the classroom in one sense but is flagged as it is inappropriate in another sense) is less harmful than providing said word and potentially leading to the retrieval of inappropriate resources for children. Hence, we discarded from our dictionary, all the terms that exist in the sexually explicit and hate-based word lists. In total, our dictionary is comprised of 60,847 unique words.

4.1.2 Phonetic Encoding Approach

Our approach to finding phonetically similar candidates is similar to that seen in the SoundEx model described by Croft et al. [10]: words are encoded to produce a phonetic key that groups those words with ones that are similarly pronounced. However, our phonetic key encoding takes inspiration from the Metaphone algorithm [50] to produce smaller groupings with less general phonetic representations. For example, our encoding differs in that the letters F and V are not considered the same sound resulting in words like *fan* and *van* being encoded to different phonetic keys. Similar is the case with Q and K, resulting in *quail* and *kale* being encoded to different keys. Although those letter pairs can make similar sounds, we found in our model development process that it was not common for children to use one of the letter pairs instead of the other. In general the phonetic encoding includes common phonetic rules, such as the letter sequence *ph* makes the /f/ sound and the *k* in letter sequences starting with *kn* is silent. While vowels are used to determine the sounds of surrounding letters (e.g., *c* followed by *i*,*e*, or *y* makes the /s/ sound), they are

removed from the final key. This is due to their ambiguity as well as preventing key groupings that are too small, resulting in several keys that only match a single word. The full phonetic ruleset for our encoding can be seen in Table 4.2. Words and misspellings are modified by each rule in the ruleset in the order shown and the result is the final phonetic form.

Table 4.2: Ruleset used to transform a word or spelling error into a phonetic key. Common word endings consist of *s*, *ing*, *ings*, and *ed*.

Step	Rule
1	Convert ‘cc’ to ‘K’
2	Replace consecutive duplicate consonants with a single consonant (e.g., ‘bb’ to ‘B’)
3	Convert ‘ck’ to ‘K’
4	Convert ‘ocea’ at the start of a word to ‘A2’
5	Convert vowels at the start of a word to ‘A’
6	Convert ‘gn’, ‘kn’, or ‘pn’ at the start of a word to ‘N’
7	Convert ‘wr’ at the start of a word to ‘R’
8	Convert ‘x’ at the start of a word to ‘S’
9	Convert ‘wh’ at the start of a word to ‘W’
10	Convert ‘gh’ at the start of a word to ‘G’
11	Convert ‘rh’ at the start of a word to ‘R’
12	Convert ‘sch’ at the start of a word to ‘SK’
13	Convert ‘y’ at the start of a word to ‘Y’
14	Convert ‘mb’ at the end of a word or before a common word ending to ‘M’
15	Convert ‘th’ to ‘0’
16	Convert ‘ch’ or ‘tch’ to ‘1’
17	Convert the t in ‘ture’ or ‘tual’ to ‘1’
18	Convert ‘sh’ to ‘2’
19	Convert the ‘c’ in ‘cion’ or ‘ciou’ to ‘2’
20	Convert the ‘t’ in ‘tian’, ‘tion’, or ‘tious’ to ‘2’
21	Convert the ‘s’ in ‘sian’, ‘sion’, or ‘sious’ to ‘2’
22	Convert the ‘c’ in ‘ci’, ‘ce’, ‘cy’, ‘sci’, ‘sce’, or ‘scy’ to ‘S’
23	Convert remaining ‘c’ to ‘K’
24	Convert ‘dge’ to ‘J’
25	Remove ‘gh’ if the next letter is a consonant
26	Remove ‘gh’ at the end of a word or before common word ending
27	Convert remaining ‘gh’ to ‘G’
28	Convert ‘gn’ at the end of a word or before a common word ending to ‘N’
29	Convert ‘y’ at the end of a word to ‘Y’
30	Convert ‘ph’ to ‘F’
31	Remove ‘h’ if before vowel, end of word, or common word endings
32	Remove ‘w’ if before consonant, end of word, or common word endings
33	Convert ‘z’ to ‘S’
34	Remove remaining vowels, convert remaining consonants to capital

To illustrate the phonetic encoding, consider the word *creature* which would be processed as follows:

- *t* in *ture* makes the *CH* sound (encoded as 1), transforming the word to *crea1ure*
- *c* is not followed by *i*, *e*, or *y*, so it makes the *K* sound, resulting in *Krea1ure*
- Remove vowels, with the exception of *Y* at the end of a word, resulting in the final phonetic form: ***KR1R***

As a pre-computational step, this process is performed on every word in the previously described dictionary and a hash table is created that maps from the phonetic key to a list of words that match that key. For example, the key ***NTRL*** maps to a list containing the words *natural*, *neutral*, and *notarial*.

To produce a larger assortment of spelling correction candidates we then use Levenshtein distance on the key of the given misspelled word to find similar keys. For example, given the misspelled word *talbe* (intended to be *table*), we would take the encoding of the misspelling, ***TLB***, and generate encodings that are 1 edit distance apart (***TBL***, ***TLBR***, ***DLB***, etc.). Despite that the intended word *table* has a different key (***TBL***) than the misspelled word, this allows to quickly find it and add to the pool of candidates. If the amount of words produced by keys of 1 edit distance from the original does not meet the amount of requested words, we then generate keys at an increasing amount of edit distance until the amount of requested words is met. Candidates are returned in ascending order of their edit distance between the keys and secondarily by their frequency in the English language.

4.2 Candidate Ranking

Once we have a suitable list of spelling correction candidates, it is essential that we rank them appropriately such that the intended word is ranked towards the top of the list. The phonetic key of spelling errors may match several different words and it's possible the key of the intended word does not match the spelling error. For this reason, we re-rank our spelling suggestion candidates with a candidate ranking system using additional informative features.

The candidate ranking system is is inspired by the work of Fomin et al. [22]. In similar work, edit distances have been used in probabilistic and machine learning methods [7, 36, 12, 36]. However, it has been shown that edit distance methods are ineffective at correcting children's spelling errors [14]. Phonetic similarity techniques, on the other hand, are more capable of correcting the spelling errors children make [14, 13] and as such we use feature sets that instead consider phonetic similarity to improve their effectiveness.

We create a feature extractor that takes in two strings: the original incorrectly spelled word and the suggested spelling. It then returns the following features which are then used to rank spelling suggestion candidates:

1. The difference in length between the suggestion and the misspelling. This featured showed promise in [22].

2. Levenshtein distance between the suggestion and misspelling. This is a traditional spellchecking strategy.

3. Frequency of the suggestion in Simple Wikipedia articles. Word frequency has been shown to be an effective method for ranking spelling suggestions [48].

4. Interpolated Kneser-Ney n-gram langauge model created from Simple Wikipedia articles. N-gram language models were shown to be a highly important feature for spellchecking in the work presented in [22]. Kneser-Ney has shown to be an effective smoothing technique [27].

5. Age of Acquisition (AoA). AoA research provides us with the age words are typically learned which is likely important when providing words for children [40].

6. Levenshtein distance between the phonetic codes of the suggestion and misspelling as determined by the KidSpell phonetic algorithm. This can tell us how similar words are phonetically.

7. Levenshtein distance between the phonetic codes of the suggestion and misspelling as determined by the SoundEx phonetic algorithm. Another phonetic algorithm inspired from the work in [22].

8. Whether or not the first letter of the KidSpell phonetic codes match between the suggestion and the misspelling. Some spellcheckers often assume the first letter in a misspelling is correct, which it often is [48], this instead looks at the first sound.

9. Number of corrections where a letter has an incorrect number of consecutive repetitions (e.g.,$ammmmaaaaazing -> amazing$: 2 corrections). This is a type of error children are known to make [16].

10. Number of unique consonants (i.e., number of consonants that appear either the suggestion or misspelling, but not both). Previous findings have shown

that vowels were not necessary when identifying the correct spelling from a given misspelling [14].

11. Number of unique vowels between the suggestion and the misspelling. In contrast to the feature above, this looks for similarity between vowels used.

Some features had high correlation with others and were removed from the final feature set. This included Metaphone[50] phonetic codes, matching of the first letter of SoundEx codes, and weighted Levenshtein distance metrics that did not penalize transpositions or substitutions of keyboard-adjacent letters and vowels.

Up to 50 candidates are generated for each misspelling then features are extracted on each. After features are extracted, we follow a similar approach to the one described in [22]. For each misspelling, we have up to 1 possible correct suggestion (assigned the value 1) and many incorrect suggestions (assigned the value 0). These are then transmitted to a machine learning model (logistic regression in the case of [22]). However, as the feature vectors described above may not be linearly separable, we also consider other models that can more effectively separate non-linear problems or provide us with more information on the importance of each of the features. This includes a decision tree, random forest, and a multilayer perceptron (MLP).

Given that providing suitable spelling suggestions in an interactive spellchecker is truly a ranking problem rather than a classification problem, as well as the behavior shown in children to have a propensity to interact with higher ranked alternatives [33, 2], we also use the learning-to-rank (LTR) model LambdaMART [8]. While, the use of LTR models has not knowingly been explored for spellchecking, they have proven effective at similar ranking problems such as large scale search, query suggestions, and recommendation [52]. While there are many LTR algorithms to

choose from, LambdaMART is regarded as one of the best-performing LTR algorithms and has been used for effectively ranking query suggestions, a problem similar to spelling suggestions [54, 52]. Preliminary evaluations of the mentioned machine learning models were evaluated and we found that the lambdaMART LTR consistently outperformed others (evaluations are reported in Appendix A). As such LambdaMART is the re-ranking model used in KidSpellλ.

Model Training

The LambdaMART model is trained to maximize on a specific metric and for this we use Mean Reciprocal Rank (MRR) with a max position of 5. As there is only 1 correct suggestion amongst many suggestions, MRR is a suitable metric for this task. A max position of 5 gives no value to items ranked outside the top 5, which prioritizes ranking suggestions within at least the top 5. Evaluations commonly look at either the top 5 or top 10 suggestions returned by spellcheckers. Given that a child audience must be considered and children tend to favor higher ranked alternatives [2, 15], we favor placing suggestions within the top 5. K-Fold cross validation and grid search was used to find the best hyperparameters for LambdaMART, which are listed as follows:

Max Depth: 5

Learning Rate: .1

N Estimators: 50

Minimum Split Samples: 2

Minimum Leaf Samples: 1

The LambdaMART model accepts a relevancy or target value for each set of items for the training process. For this task, the one correct suggestion is given a value of 1 and all other incorrect suggestions are given a value of 0. We refer to this improved ranking model as KidSpellλ.

CHAPTER 5

SPELLING CORRECTION EVALUATION

In this chapter we evaluate the effectiveness of our method in two areas: the first being the *candidate generation* of KidSpell and the second being *candidate ranking* of KidSpellλ. Both are evaluated on spelling errors generated by children described in Chapter 3. We also examine the relative feature importance for the features described in Section 4.2.

5.1 Candidate Generation

The experiments in this section examine both the effectiveness of the efficiency of KidSpell's candidate generation method against state-of-the-art methods. KidSpell uses a phonetic algorithm to generate candidates of matching keys as described in Section 4.1.

5.1.1 Method

Each candidate generation method (KidSpell's and the baselines) is asked to generate candidates for each spelling error in the $\textbf{Childrens}_{\textbf{MSP}}$ dataset. We vary the number of k candidates generated from each method because they directly influence the time complexity and the search space reduction.

5.1.2 Baselines

The most similar work to ours for generating suitable candidates is the work done by Pande [49] and de Amorim and Zampieri [12]. As the work performed by Pande is the most recent and shows a significant improvement to the results found by de Amorim and Zampieri, we use their work as a baseline. Pande uses neural character embeddings that use character sequences that are generated using consecutive vowels or consonants, but not both (e.g., 'affiliates' generates 'a ff i l ia t e s'). Our implementation uses size 100 embeddings as that provided the best performance. We also used a modified version of their algorithm that instead uses single character sequences used in character embeddings, which performed better with children's misspellings (e.g., 'affiliates' generates 'a f f i l i a t e s'). All methods used the same dictionary for candidate generation.

5.1.3 Metrics

We report on the *success rate* which is the percentage of spelling errors for which the gold standard (intended word) is among the pool of suggestions generated. As the goal of these methods is to reduce the time complexity of spelling correction algorithms by limiting the search space, we also measure the runtime (i.e., seconds to generate the k candidates for all words in the entire $\mathbf{Childrens_{MSP}}$ dataset).

5.1.4 Results

The success rates of the *KidSpell* phonetic algorithm and the two baselines (labeled *Pande Embeddings* and *Character Embeddings*) are presented in Figure 5.1. The KidSpell phonetic algorithm significantly out performs the two baselines for every

variation of k candidates using a paired t-test ($p < 0.05$; n=1785). Most noticeably it outperforms significantly on the lower end of k variations. Just generating 100 candidates with the *KidSpell* phonetic algorithm outperforms both baselines even when given the opportunity to generate 5000 candidates. Notably, the embeddings methods had difficulties picking up spelling errors that were more than just a couple edits away (e.g., *favtit* for *favorite* requres 3 single character edits). They also tended to return substrings of the misspelling that matched a real word (e.g, returning *since* for the misspelling *sincerly*). The KidSpell phonetic algorithm benefited from returning words in order of frequency as the embeddings had a tendency to return obscure words.

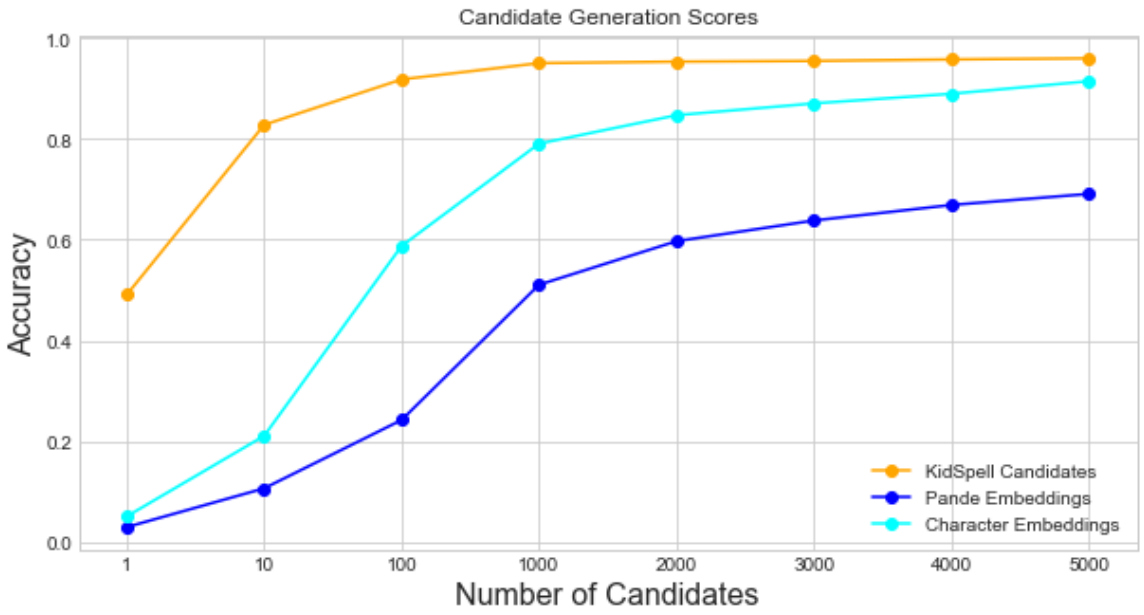

Figure 5.1: Success rates (%) for various k (number of candidates)

The runtime of the various algorithms are reported in figure 5.2. The *KidSpell* phonetic algorithm performs more efficiently when generating a lesser amount of candidates, but falls slightly behind when generating 1000 or more candidates. When

combined with the findings of the success rates, the KidSpell phonetic algorithm can effectively find a more precise and smaller candidate pool in a fraction of the time. For example, 100 KidSpell candidates is more likely to contain the correct suggestion than any of the baselines at 5000 candidates and can be generated in just a sixth of the time it takes the others to generate a less effective pool of 5000 candidates. In fact, generating more than 100 candidates using KidSpell's phonetic algorithm becomes unnecessary as we nearly achieve our peak performance.

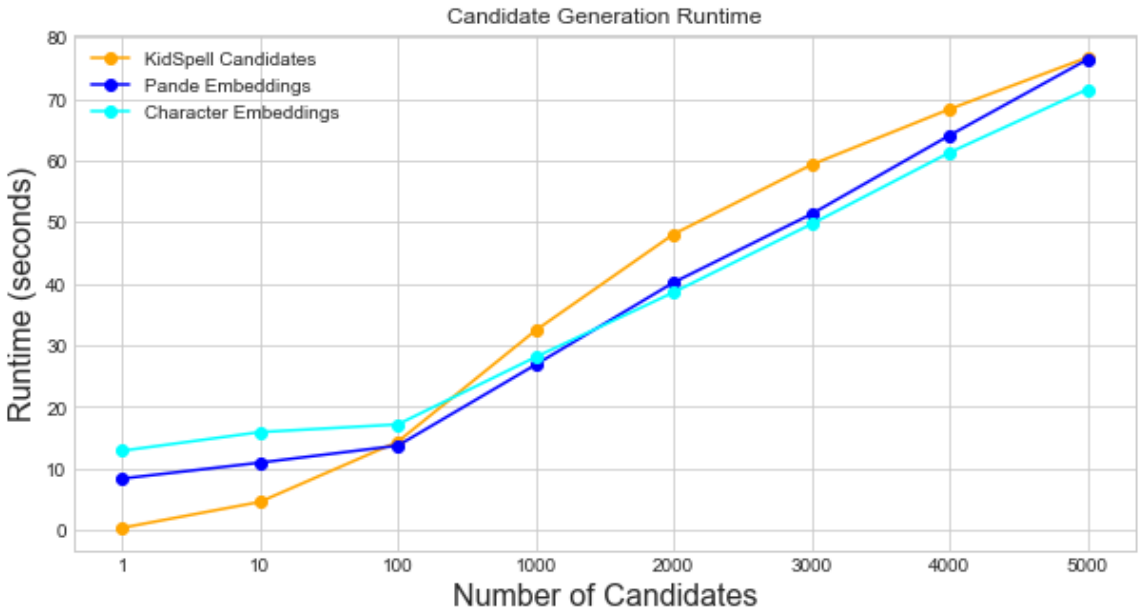

Figure 5.2: Runtime in seconds for various k (number of candidates)

5.2 Spelling Correction

In this section we evaluate the fully fledged **KidSpell**λ spellchecking model, assisted by the candidate generation demonstrated in the previous section, against other state-of-the-art spellcheckers.

5.2.1 Method

All spellchecker methods (KidSpellλ and other described below) are evaluated on the **Childrens$_{\mathbf{MSP}}$** dataset described in Section 3. This includes both the spelling errors made in hand-written essays as well as typed search queries. Metrics from models requiring training (such as the KidSpellλ model described in Section 4.2) are taken as an average from a 5-fold cross-validation (80% training, 20% test).

5.2.2 Baselines

Several baselines were chosen to compare to our method, these include:

- **Aspell (Normal)** - Aspell was chosen as a common spellchecking baseline [7, 13] that was also the best performing baseline in our previous research on correcting children's spelling errors [14]. Aspell also utilizes phonetic encodings (Metaphone algorithm [50]) in a similar manner to our approach making it potentially effective for children's spelling errors.

- **Aspell (Bad Spellers)** - Aspell with *Bad Spellers* mode enabled was chosen as the goal of correcting the spelling of bad spellers is the most similar to our work.

- **Bing** - Microsoft's Bing Spell Check API was used as an industry standard for correcting spelling in the search context as well as being a popular search engine preferred by children [23].

- **KidSpell** - The final baseline is the KidSpell candidate generation method without the improved ranking as described in Section 4.2.

Each baseline uses the dictionary supplied with the software. Both verisons of KidSpell use the same dictionary. The methods are all compared to the improved KidSpellλ model.

5.2.3 Metrics

To measure the performance of the respective spellcheckers we use Hit Rate and Mean Reciprocal Rank (MRR).

Hit Rate measures the rate at which the intended word (gold standard) appears in the list of spelling suggestions. For each spelling error in the dataset, the spellchecker receives a value of 1 if the gold standard is in the list of spelling suggestions, otherwise this value is 0. An average is taken to determine Hit Rate.

MRR measures how well ranked the spelling suggestions are by capturing the average position of the relevant spelling suggestion. This is measured with the following equation:

$$MRR = \frac{1}{|C|} \sum_{i=1}^{|C|} \frac{1}{rank_i}$$

where C is the set of spelling errors, $|C|$ is the number spelling errors, and $rank_i$ is the ranking position of the of the gold standard. A higher MRR value indicates a higher average ranking for the gold standard. Given children's propensity to click on higher ranked alternatives for spelling suggestions [15] as well as other areas of search [2, 33], it is crucial to rank the gold standard highly. Taken together, these two metrics measure how well each spellchecker is at finding the intended word (Hit Rate)

and how well it is ranked (MRR). Numbers for hit rate are reported for a varying number spelling suggestions (k 1-5) and MRR is reported using the top 5 suggestions.

5.2.4 Results

Results of the hit rate of KidSpellλ along with the baselines on the full **Childrens$_{\mathbf{MSP}}$** dataset are described in Figure 5.3. Note that the *Bing Spell Check API* only returns a maximum of 3 suggestions per spelling error, limiting its performance when k is greater than 3. The reported KidSpellλ results are significantly better when compared to the baselines intended for adult users as well as the original KidSpell method using a paired t-test ($p < 0.05$; n=1785). While Aspell's Bad Spellers mode does provide a minor increase over the normal Aspell, it still has difficulties when dealing with children's spelling errors. Even just providing 1 suggestion from KidSpellλ is more likely to provide the gold standard than 5 suggestions from the best alternative (Aspell Bad Spellers mode).

Results for the MRR scores for each spellchecker are provided in Figure 5.4. The improvement for KidSpellλ is statistically significant using a paired t-test over all alternatives ($p < 0.05$; n=1785). These scores indicate that KidSpellλ is able to include the gold standard within the first 2 suggestions on average, while the alternatives provide the gold standard at the 3^{rd} position on average.

We further examine the results of the spellcheckers on the two different environments they were created in (hand-written essays and typed search queries). The hit-rate and MRR for spelling errors made in hand-written essays are reported in Figures 5.5 and 5.6 respectively. Given that the large majority of the **Childrens$_{\mathbf{MSP}}$** is made of hand-written essay spelling errors (1651 out of 1785) we see similar results

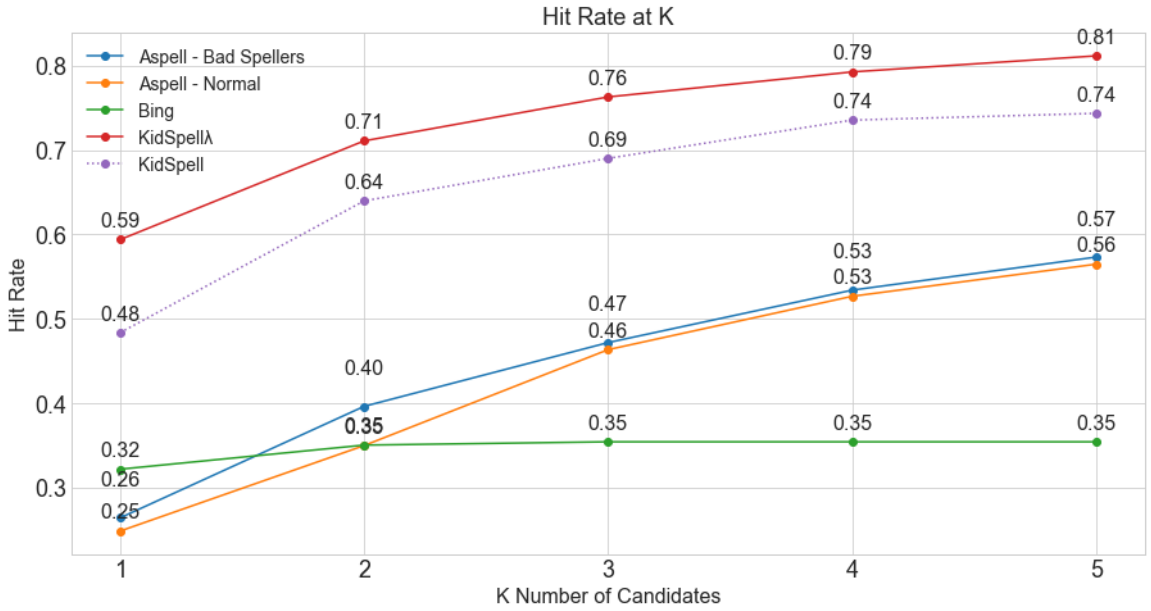

Figure 5.3: Hit-Rate for various k (number of suggestions). Note that Bing is limited to 3 suggestions.

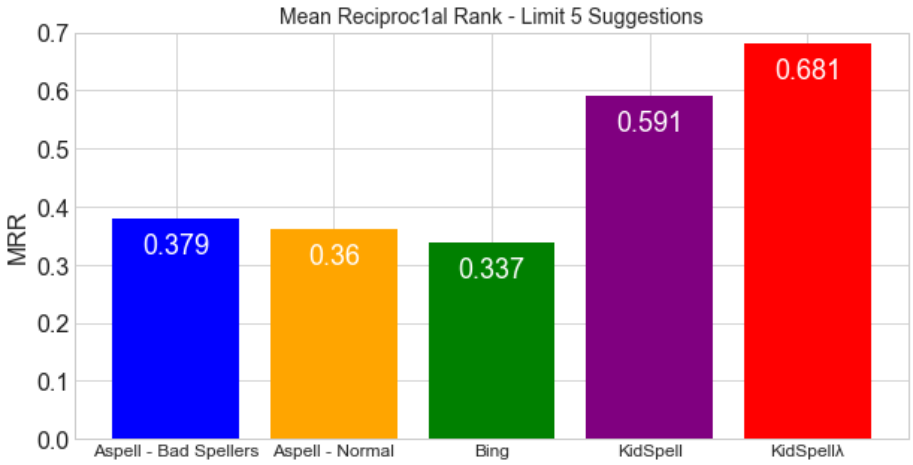

Figure 5.4: MRR using top 5 suggestions

to those of the full dataset. Likewise, the improvement for KidSpellλ is statistically significant for both the hit-rate and MRR ($p < 0.05$; n=1651).

The hit-rate and MRR for spelling errors made in the typed search queries are reported in Figures 5.7 and 5.8 respectively. Here, all but Bing Spell Check perform

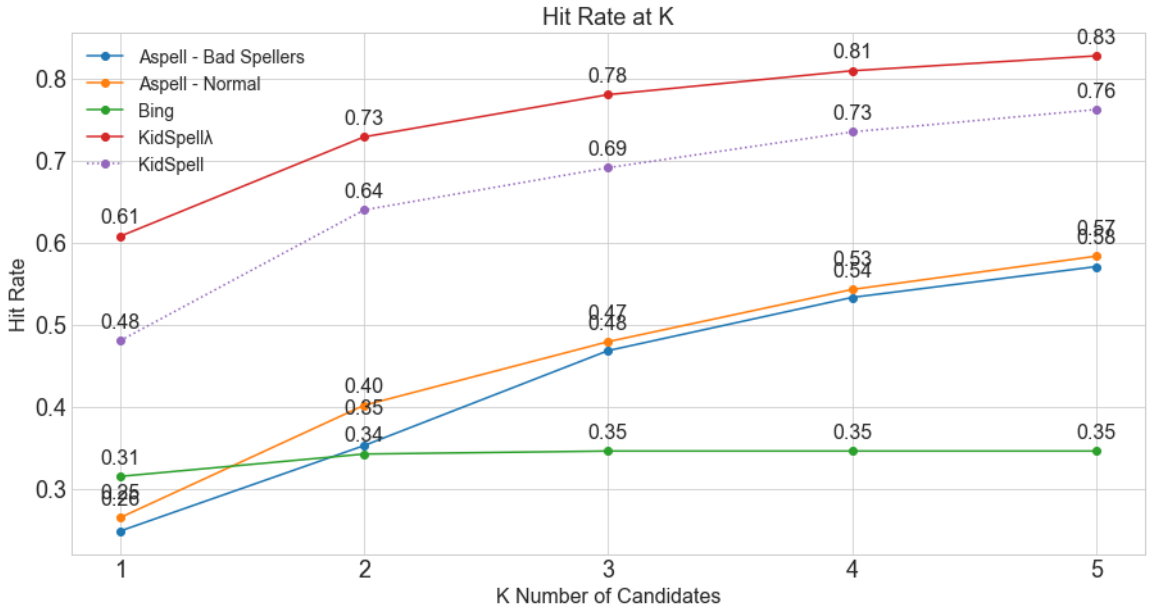

Figure 5.5: Hit-Rate for various k (number of suggestions) on spelling errors made in hand-written essays

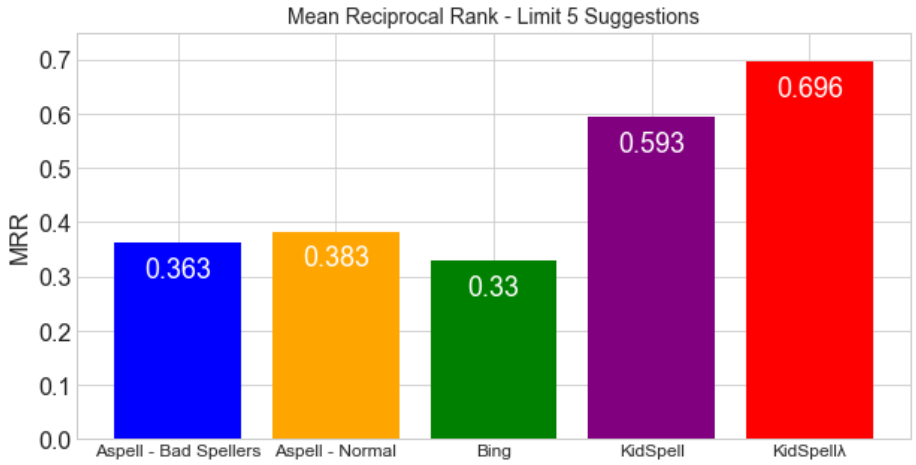

Figure 5.6: MRR using top 5 suggestions for hand-written essays

considerably worse than on the hand-written essay spelling errors. We attribute this to the *mistyping* errors that can occur in while using a keyboard. Since KidSpellλ and Aspell both rely on phonetic information, mistyping errors such as *tghat* for *that* make the words seem like unlikely matches since the misspelling and the gold

standard do not match phonetically. Similarly, typed spelling errors are much more likely to include *boundary errors*. These errors consist of including a space when there shouldn't be one (e.g., *com puter* for *computer*) or the lack of a space when there should be one (e.g., boisestate for *boise state*). These types of errors are overlooked by KidSpellλ. Bing Spell Check is better at handling these types of errors and noticeably performs better when being used for its intended purpose (i.e., correcting misspelled queries) however it still falls behind in performance when compared to KidSpellλ. KidSpellλ's improvement over both the original method and the alternative is statistically significant when providing 5 suggestions using the paired t-test ($p < 0.05$; n=134). It is also worth nothing that although KidSpellλ was trained on primarily hand-written essay spelling errors, it still provides a significant improvement over the original KidSpell method when handling typed query errors.

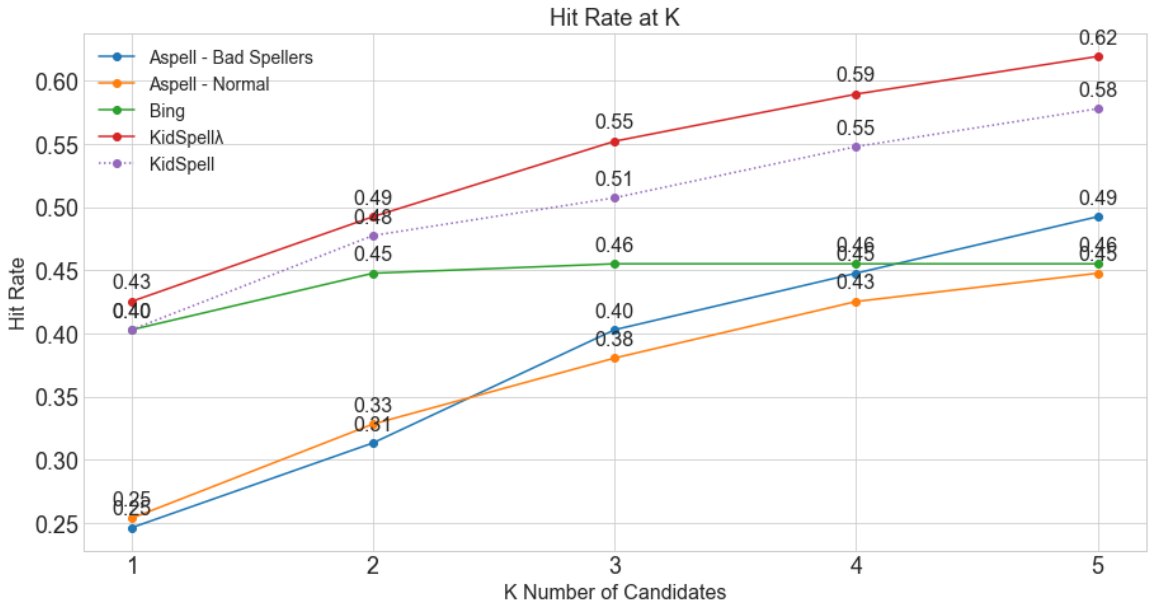

Figure 5.7: Hit-Rate for various k (number of suggestions) on spelling errors made in typed search queries

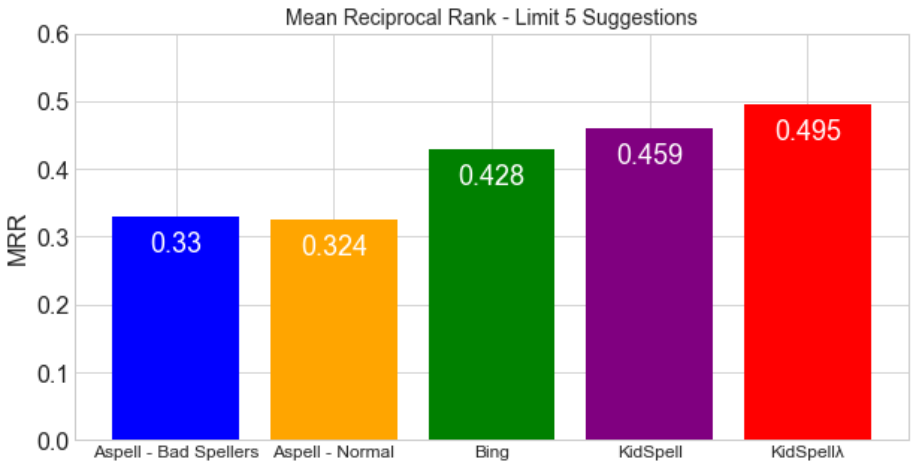

Figure 5.8: MRR using top 5 suggestions for typed search queries

Grade Levels, Spelling Levels, and Native Languages

To further analyze KidSpellλ when compared to other spellcheckers, we evaluate their ability to correct spelling errors from children of different grades, spelling ability, and native languages which are included as part of the hand-written essay spelling errors as described in Section 3. As such, all spelling errors in this section are all a subset of the hand-written essay spelling errors dataset ($\mathbf{Essay_{MSP}}$).

Evaluations on the hit-rate at 5 of the spellcheckers on students grades K through 8 are included in Figure 5.9. While most spellcheckers see a general trend upward in hit-rate as children's grade level increases, KidSpellλ experiences the least variation while maintaining a higher hit-rate at all grade levels. The dip for most spellcheckers at grade 7 can be explained by the limited data and number of students at that grade level. The amount of spelling errors and students for each grade level are included in Table 5.1.

As grade level is not necessarily indicative of a child's spelling ability, we also include evaluations on the hit-rate of the various spellcheckers separated by the spelling

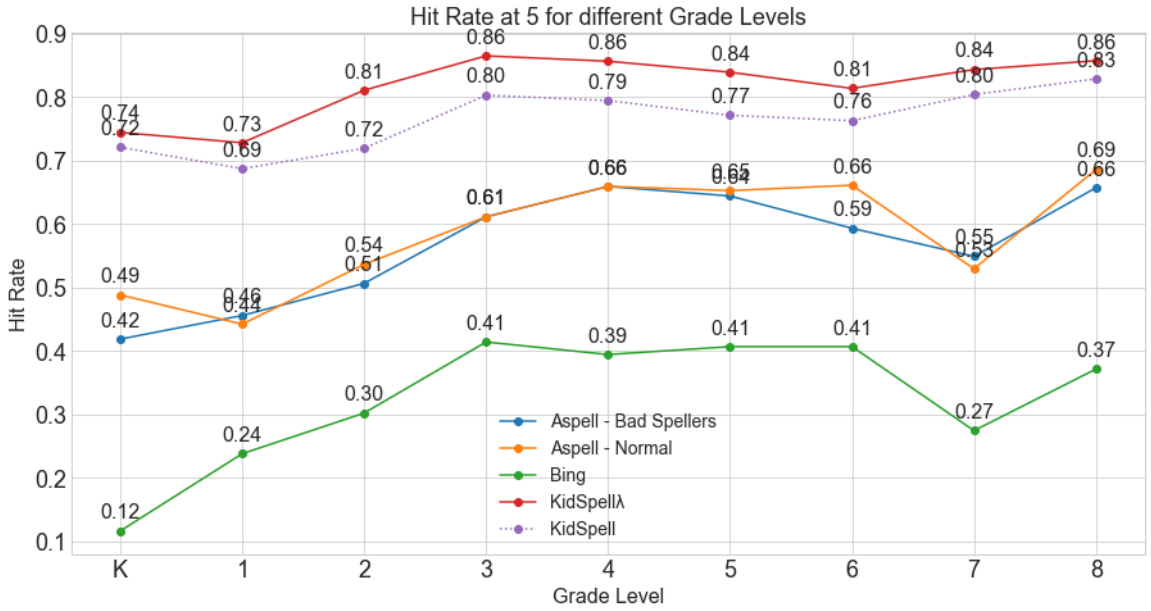

Figure 5.9: Hit-Rate at 5 for spelling errors made by children in grades K through 8

developmental level of the students as described in Section 3. The hit-rate at 5 is included in Figure 5.10. A table of the number of spelling errors and students for each development level is included in Table 5.2. Similar to the grade levels, spellcheckers see a trend upward as spelling level increases. This is especially noticeable for the adult-oriented baselines. The closer the spelling level of the student gets to the intended audience (i.e., adults), the better they perform. Meanwhile, KidSpellλ is able to keep a hit rate of 80% or higher regardless of spelling development level. The results seen for the different grade levels and spelling levels emphasizes KidSpellλ's importance for especially young or new spellers. The dips for the derivational spellers

Table 5.1: Number of spelling errors for each Grade Level K through 8

Grade	K	1	2	3	4	5	6	7	8
Errors	43	147	470	355	355	118	59	51	35
Students	6	15	29	22	16	9	5	4	4

and inconsistencies for emergent spellers can be explained by the limited amount of data for each of those groups.

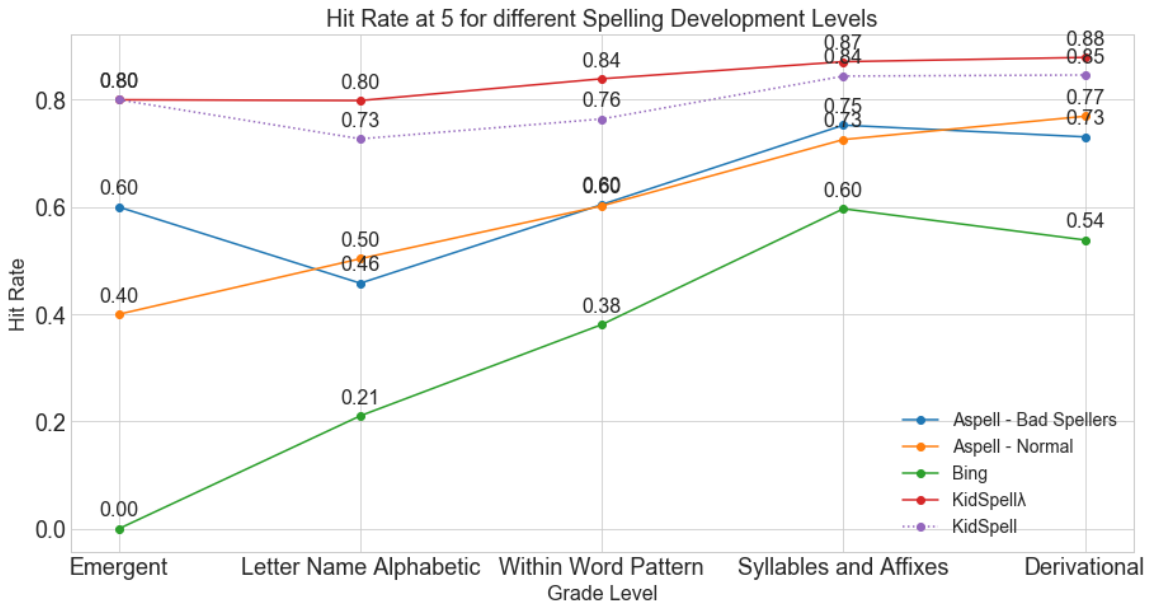

Figure 5.10: Hit-Rate at 5 for spelling errors made by children at different spelling developmental levels

Table 5.2: Number of spelling errors for each spelling development level

Level	Emergent	Letter Name	Within Word	Syl.&Aff.	Deriv.
Errors	5	546	827	186	26
Students	2	25	45	16	6

For a partial amount of the hand-written data, children's native language was recorded. We examine the hit-rate at 5 of the different spellcheckers for each lan-guage as well as for english vs non-english in Figures 5.11 and 5.12 respectively. As the phonetic information for both KidSpellλ and Aspell are both based in English we might expect that they perform worse when handling spelling errors made by children that speak a different native language. While KidSpellλ is able to perform significantly better than the baselines for each individual language (paired t-test; p

< 0.05), there is no significant difference between KidSpellλ's performance on native English spelling errors and KidSpellλ's performance on native Non-English spelling errors (2 sample t-test; $p > 0.05$). Counts of spelling errors and number of students those spellings errors were generated from for each native language are reported in Table 5.3. Note that Japanese and Mandarin are noticeable outliers with just 16 and 2 spelling errors of each respectively. Among other languages, KidSpellλ is consistent in achieving a hit-rate of at least 80%. Other spellcheckers seemed to struggle more when handling spelling errors from Spanish or Italian native speakers.

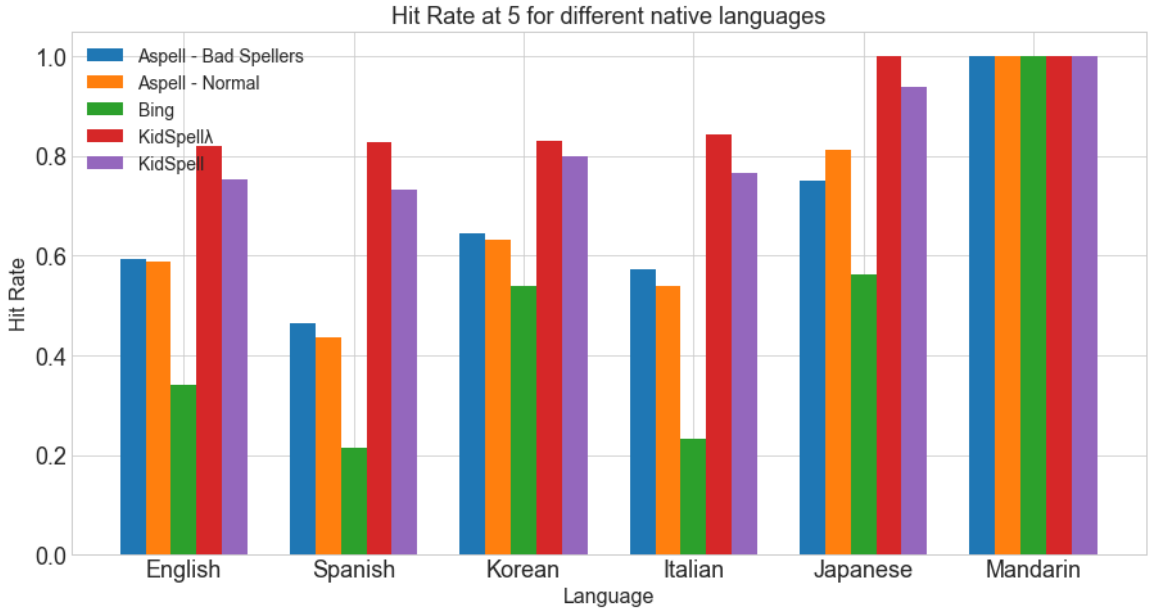

Figure 5.11: Hit-Rate at 5 for spelling errors made by children with different native languages

Table 5.3: Number of spelling errors and students for each native language

Language	English	Italian	Spanish	Korean	Japanese	Mandarin
Errors	942	209	264	197	16	2
Students	42	2	2	30	2	1

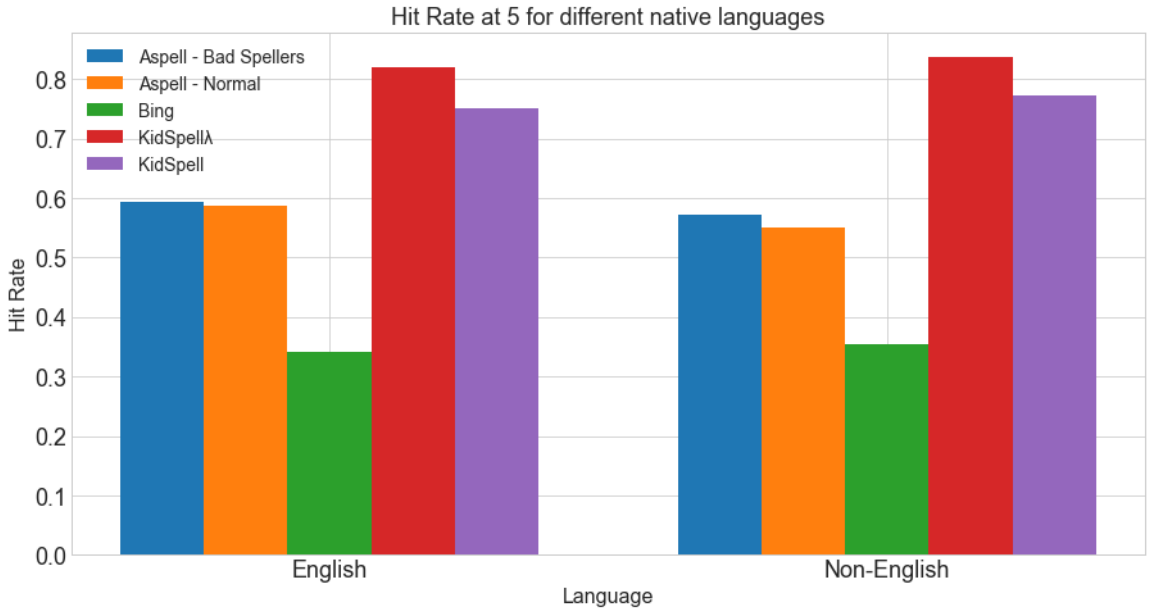

Figure 5.12: Hit-Rate at 5 for spelling errors made by children with english and non-english native languages

Feature Importance

The features importance for each of the features listed in Section 4.2 for the lambdaMART LTR model are listed in Figure 5.13. We also include the feature importance for other tested models: logistic regression, decision tree, and random forest classifiers are reported in figures 5.14, 5.15, and 5.16 respectively. The higher the value, the more important the feature is. In all but the logistic regression, the edit distance of the KidSpell phonetic algorithm is considered the most important feature. For the lambdaMART LTR, it is by far the important component of the feature set. This emphasizes the importance of phonetic information when correcting children's spelling errors. Edit distance of the SoundEx phonetic algorithm on the other hand scores low. While they did not have a high correlation, they do fill similar rolls and the KidSpell phonetic algorithm may provide much more precise information.

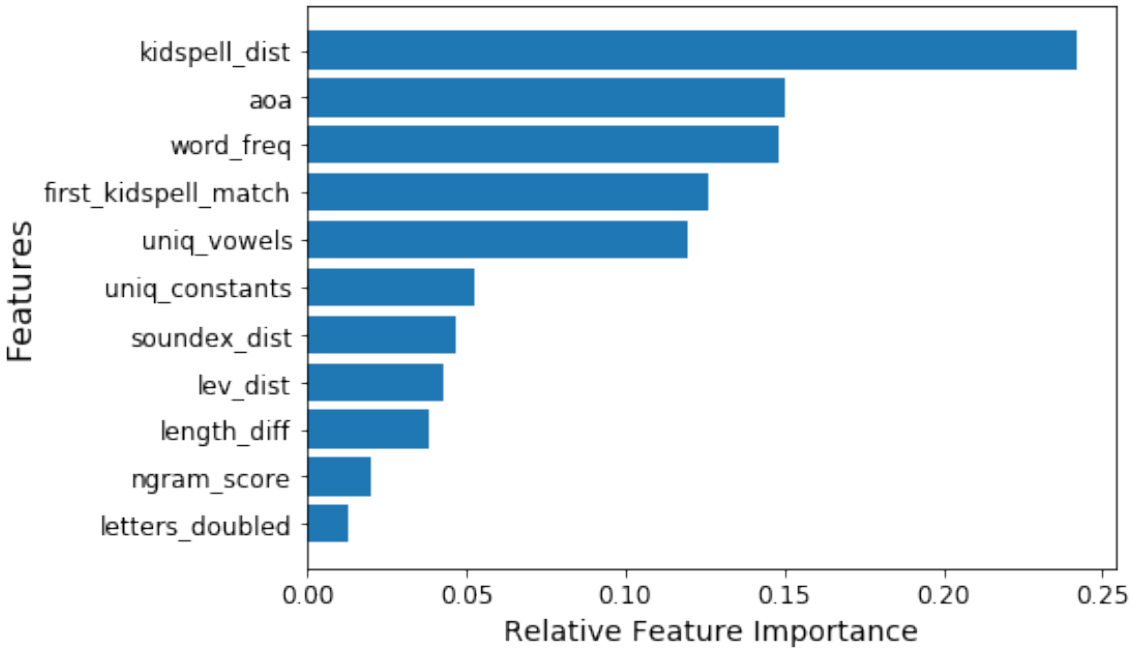

Figure 5.13: Feature importance using lambdaMART LTR

Age of acquisition and word frequency are second and third for lambdaMART in feature importance and are found to be important features for the other models as well. Given that children are more likely to know or use more frequent words or words within their age of acquisition [40], it's unsurprising that the models value these two features. Unique number of vowels is another highly important feature for lambdaMART. This feature is surprising as the KidSpell phonetic algorithm was built around children's inconsistent use of vowels and ignores them all together. Perhaps, the lack of information of vowel usage that we're getting from the KidSpell phonetic edit distance puts higher value in this feature.

Other than on the random forest classifier, n-gram score importance was low. Related work using n-gram scores for adult spellings were considered highly important [22]. Unsurprising Levenshtein distance was typically unimportant. This aligns with

our findings that spellcheckers that use Levenshtein edit distance do not perform well when correcting children's spelling errors [14].

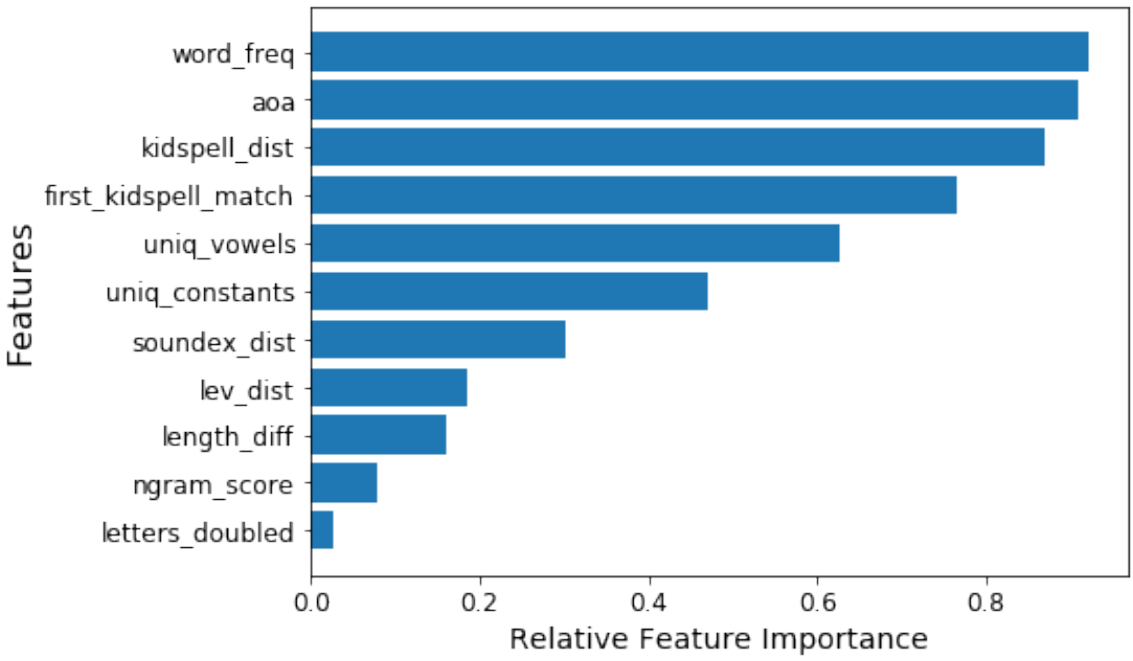

Figure 5.14: Feature importance using logistic regression classifier

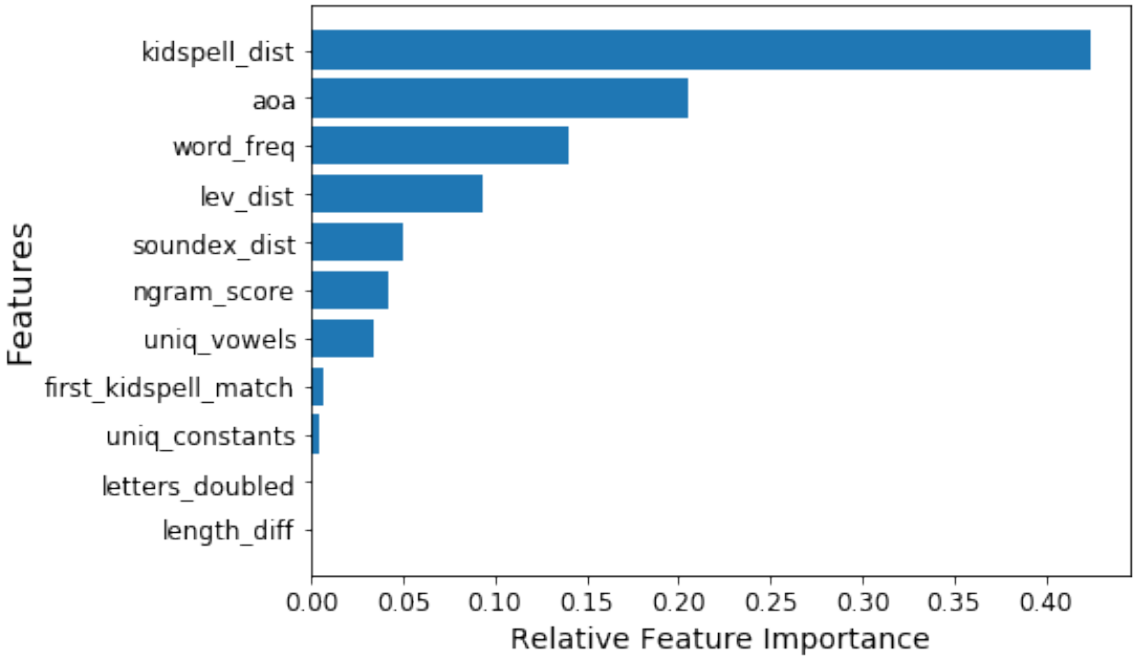

Figure 5.15: Feature importance using decision tree classifier

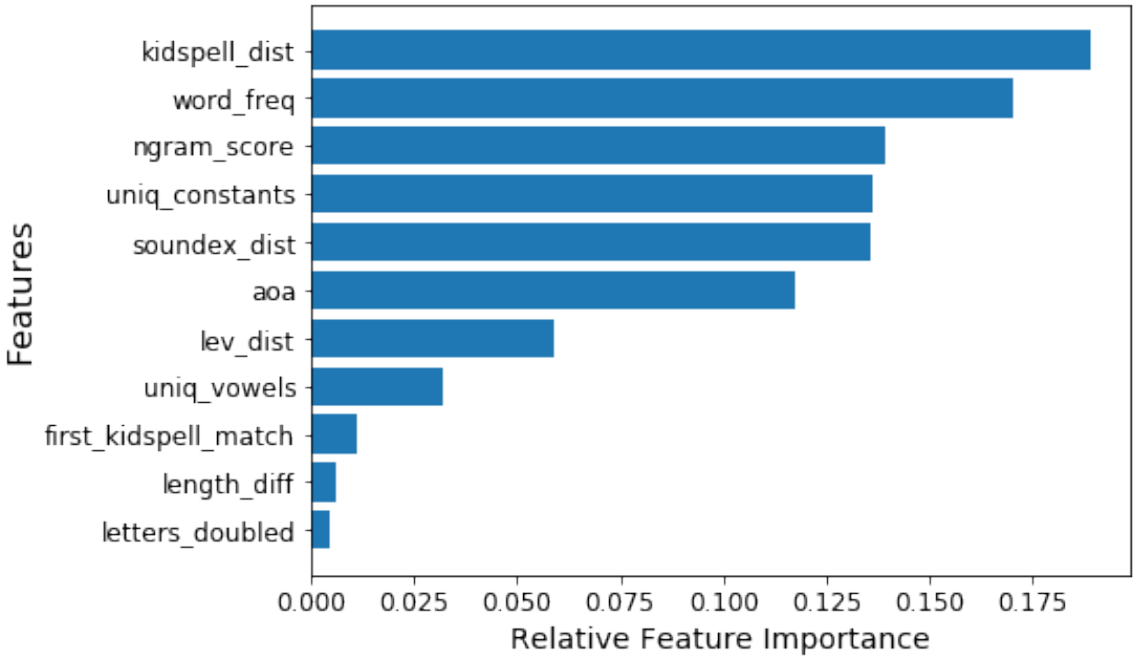

Figure 5.16: Feature importance using random forest classifier

CHAPTER 6

SPELLCHECKING INTERFACE

In this chapter we discuss the visual and audio cues explored to help children identify and select their intended spelling suggestions. We then discuss the participatory design sessions completed in an effort to help users better notice and respond to spelling errors.

6.1 Multimodal Cues for Spelling Suggestions

Despite advances demonstrated in spelling correction for children, we cannot expect algorithms to fully recover the intent of the child as the first spelling suggestion, which is a reason why spellcheckers provide multiple suggestions. This led to investigating the effect multimodal cues (i.e., images and audio playback) can have on helping children select the word that best matches their intent [15]. While a child-oriented spellchecker may better respond to children's spelling errors, children's behavior to gravitate towards higher ranked spelling suggestions [14], even if the word does not match their intent, can impedes its effectiveness. To enhance spellchecking functionality, we incorporated multimodal cues in an effort to investigate how these cues can help children effectively select the spelling suggestion that meets their original intent. In this section we summarize our findings for the inclusion of multimodal cues for spelling suggestions.

We conducted a between-subject experimental study using a custom search tool with four conditions, each with varying media cues to go along with the spelling suggestions: no cues, audio, image, or both audio and image. There were 191 child participants (age 6-12) who took part in different experiments located at local STEM events. Each child interacted with a custom search tool with one of the conditions chosen randomly.

In the custom search tool, when a word is identified as potentially misspelled, it is colored and underlined in red. Hovering over a misspelled word opens a list of spelling suggestions. When a spelling suggestion is hovered, an image is displayed and/or speech synthesis is used to read the word aloud depending on the condition. Images used in the interface were acquired using Google's Image Search API with safe search enabled. The first image returned, using the spelling suggestion as the search query, was the image chosen to be displayed alongside each spelling suggestion. Speech synthesis for the audio playback is acquired using Amazon Polly[1]. Examples of the different visual interfaces can be seen in Figure 6.1.

In order to identify which multimodal cues, if any, would better guide children's ability to select their intended word, we adapted the protocol defined in [42] to allow us to systematically compare across separate experiments. The protocol outlines four dimensions which we specify as follows:

1. **Task**. Verbal prompts were given to child participants to serve as a starting point for a typical online search query. For this, we relied on two types of prompts: *fact-based*, which are less complex and require children to locate specific and quick answers, as well as *open-ended* prompts, which require more

[1]Amazon Polly: https://aws.amazon.com/polly/

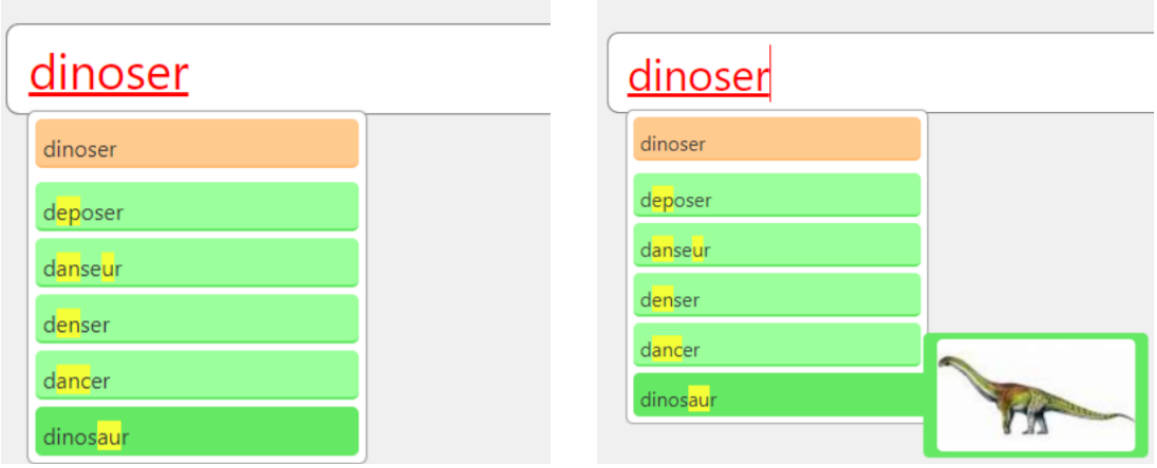

Figure 6.1: Spellchecking function without visual aids (left) and with visual aids (right). Both may be accompanied by audio playback.

in-depth consideration of the content of the search results. Sample questions can be seen in Table 3.3.

2. **User group**. Participants were children ages 6 to 12. We selected this age group to represent children who have likely developed the basic phonetic skills needed to attempt spelling, but have yet to obtain advanced orthographic skills [5].

3. **Strategy**. We make use of a custom search tool targeting children described above in which all user interactions with the interface are automatically recorded. Facilitators observed and recorded interactions children made with the search. These were used in combination to analyze children's interactions with the interface.

4. **Environment**. Search tasks were performed by children at local STEM events hosted at three local venues – two elementary schools and a local community building. The stem event held at the community building was organized by

state government agencies where children were bussed in from their respective schools and participated as an informal education (i.e., a fieldtrip) experience. Each event contained multiple STEM-related booths and hands-on activities.

In our experiments we examined how accurately children clicked on their intended word and in which position those clicked occurred. The results of these experiments are summarized in Table 6.1. We saw significant improvements to children's ability to find their intended word among a list of spelling suggestions when using either of the multimedia cues or a combination of the two. Although these improvements were shown to be of statistical significance when compared to the baseline experiment that had no cues (two-proportions z-test; $p < 0.05$), we did not observe a statistical significance when comparing them to each other (two-proportions z-test with Bonferonni correction; $p > 0.016$).

The audio only condition performed the best with the highest accurate click percentage (92%) as well as having 0 incorrect clicks in the first position and 3 incorrect clicks in the first three positions, which was half or less than any of the other experiments. Other conditions noticeably still had children resort to clicking on suggestions in the first position, even if that was not their intended word. Given that the condition with both audio and images performed worse than the audio condition suggests to us that the use of images may have had a direct impact on how useful the audio was. Why images did not have as much as an impact is an open question, but this could be explained by the fact that spelling mistakes made by children are phonological [6] and audio provides feedback on the phonetics of a word while images do not. Another possible explanation is that many words that children learn are concrete in that they denote physical objects (e.g., 'bird' or 'ball') and will likely

have images that represent them well, whereas more abstract concepts ('democracy' or 'because') do not.

Table 6.1: Analysis of spelling suggestions.

K	No Cues			Audio Only			Images Only			Both Audio & Images		
	Clicks	Correct	%	Clicks	Correct	%	Clicks	Correct	%	Clicks	Correct	%
1	28	21	.75	16	16	1.0	16	12	.75	12	9	.75
2	4	2	.50	12	10	.83	13	12	.92	14	13	.93
3	4	3	.75	19	18	.95	7	5	.71	16	14	.88
4	8	5	.63	9	9	1.0	7	7	1.0	21	19	.90
5	3	1	.33	9	7	.78	8	8	1.0	9	9	1.0
Total	47	32	.68	65	60	**.92**	51	44	.86	72	64	.89

The findings presented show that the assistance of cues in any of the conditions (audio only, images only, or both audio and images) are all a statistically significant improvement over having no cues at all. The audio only condition showed the best results in terms of both having accurate clicks and avoiding a pattern of resorting of the first available option. The results found as part of this study demonstrate the importance of having some kind of cue to go along with spelling suggestions when presenting them to children. While audio and/or image cues are helpful, they each have their drawbacks. The use of images presents issues when representing words that are not concrete in that they do not denote physical objects and may require some curation to perform optimally. Audio is not always available or appropriate in the context and the synthesized speech cannot be relied upon to always pronounce words correctly.

6.2 Participatory Design

In our studies presented in Section 6.1 we observed that not all children noticed or utilized the misspellings or suggestions without being prompted. In order to

create potential solutions to these issues, we used participatory design where child participants acted as design partners [20, 32]. This process helps us understand their needs and how to ensure we meet them. The child participants involved are members of KidsTeam, an inter-generational design team that meet twice a week. The goal of the team is for children and adults to work collaboratively as design partners to design technologies for children. Child participants vary from novice to intermediate in computer abilities.

During the participatory design sessions, child participants along with adults worked together to design a spellchecking interface with a focus on bringing attention in two facets. The first facet being how to better indicate that a word is misspelled and the second facet being on how to improve the interface to get users to click/interact with the misspelled word and select one of the suggestions.

In the first session, we introduced children to the basic spellchecking interface. The interface the children interacted with only marked spelling errors by coloring the text of the word red and underlining it. To produce spelling suggestions children would have to click or tap on the misspelled word. Children were split into groups consisting of 2-3 children and 1 adult. Each of the groups worked collaboratively to come up with ideas on how to improve the spellchecker interface without taking away from the search process. After having a chance to interact with the interface, children and adults worked collaboratively using the "big paper" participatory design technique [20].

Children came up with several ideas to improve the interface that commonly included highlighting the spelling errors red, circling the words like a teacher would on a paper, audio feedback (bell or chime sounds), and automatically displaying spelling suggestions such that a user would not have to click on the misspelling. Using red on

a word was unanimously decided as the best color to indicate that it was misspelled. More unique ideas came out of this as well that included haptic feedback (via a "vibrating keyboard"), "notifications" like you might see on a mobile device, and increasingly pronounced indicators of the misspelled words. Other ideas involved animated elements on the screen such as bells to go along with the audio feedback and an animated circle around the misspelled word (e.g., circling it). Examples of their big paper design ideas can be seen in Figure 6.2.

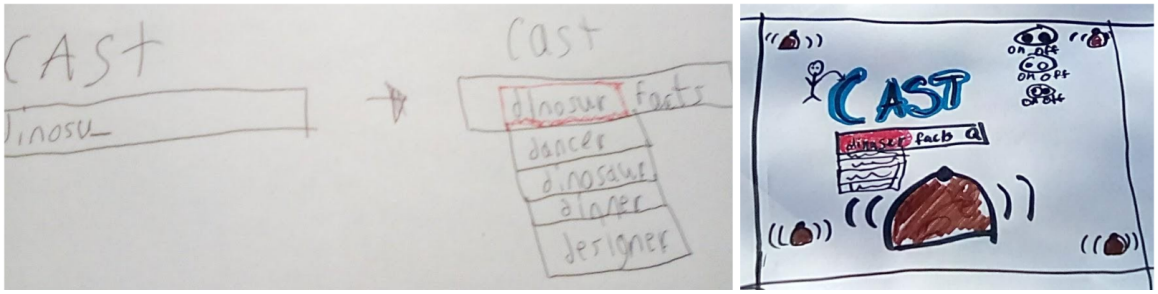

Figure 6.2: Big paper examples from the first design session

From this we improved the interface to include some of the most common and well-liked ideas. This included playing a bell or chime sound when a word was misspelled, automatically displaying the spelling suggestions without requiring tapping or hovering, and including an animation to circle the misspelled words when they appeared.

In a subsequent session, children and adults were split into groups as was done previously. However, this time we used a participatory design technique involving children recording their individual likes, dislikes, and design ideas on sticky notes [20]. Children interacted with the interface that incorporated changes reflecting many of their ideas from the previous session. The team worked together to come up with ideas on how to better display images and play sounds for each suggestion on their tablet

devices. On the standard interface, hovering over the spelling suggestion would read the word aloud using synthesized speech and display an image next to it. However, with touchscreen devices like their tablets, hovering over interface elements is not possible so the team collectively worked to solve this issue.

Children mostly liked the ideas that came out of the previous session. However, while some liked the sound that played when a misspell occurred, some did not. One idea was to use synthesized speech to alert the user of the misspelling, similar to the synthesized speech used for the spelling suggestions. One common idea included having a speaker button next to the spelling suggestions that you could tap on to play the word aloud and another button to display an image next to it. Another common idea included having the words read aloud one by one, along with their associated images being displayed which would be done automatically when a misspelling occurred. Other ideas included customization options to turn off the spellchecker, change the sounds, or an option to close the spellchecker and ignore a misspelling if it was marked incorrectly. Examples of the sticky notes created can be seen in Figure 6.3.

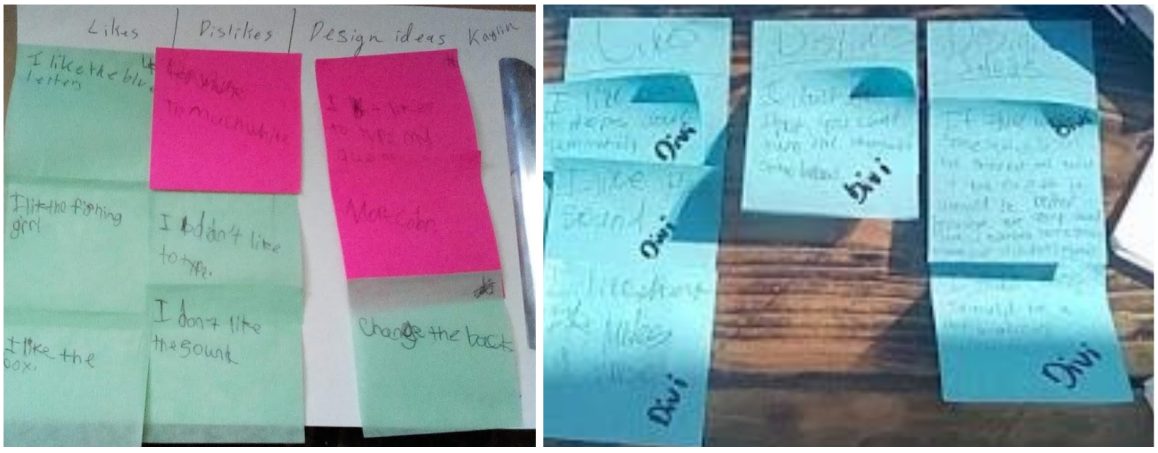

Figure 6.3: Sticky note examples from the second design session

For the next session we incorporated both the ideas of having a button next to each misspelling as well as having the suggestions read aloud automatically when a misspelling occurred. We additionally included a synthesized speech to say the phrase "Did you mean one of these?" when a misspelling occurred. For this session, we did another likes-dislikes-design ideas sticky note technique similar to the last session. Children enjoyed that it "talked" and displayed images for each suggestion. As children were more exposed to the chime sound that was played when a misspell occurred, they expressed their dislike of it. Other common dislikes included that the sounds were repetitive, pictures associated with each spelling suggestion were too small, and that the pictures often just showed a picture of the word itself. Common design ideas included changing the color of the spelling suggestion buttons to improve the contrast of the highlighted differences between words and an option to close the spellchecking suggestions.

For our last session we implemented color changes, removed the chime sounds, and implemented two separate interfaces for closing the spelling suggestions window. In one option (A), the close button was added to the button of the list of spelling suggestions that simply said "Close" and in the other option (B), the misspelling button was moved to the bottom since it served the same purpose. Children once again worked with adults to explore the new features, choose their favorite, and offer any likes, dislikes, or design ideas they might have. Nearly all children preferred option A with the separate "Close" button. Children also preferred the new colors over the older versions. A still image of the interface with the final design ideas can be seen in Figure 6.4.

To assist with the repetitive sounds that occurred when misspelling a word we included some variations of the original phrase to be chosen randomly that included

Figure 6.4: Still Image of the final interface interface after incorporating design ideas from children in participatory design sessions

"What about these?", "Is this what you mean?", etc. We additionally included some pause/cue phrases (e.g., "um","hmm", or "ok") at the beginning of phrases when the voice was interrupted because of another misspelling occurring. This follows spoken dialogue research that suggests that dialogue interruptions and resumptions should start with a various lexical cue phrase [19].

Overall, our findings in these design sessions align closely with our previous research [15] of children's attentiveness towards audio and visual cues and advanced it by addressing issues identified in the previous study. It also demonstrates some problems remaining with spellchecking. In one case, a child participant commented positively on the spellcheckers ability to find their intended word and in another they noticed that their intended word was not in the list of suggestions. While many children expressed that they liked the pictures, the method for generating pictures

did not always provide meaningful images as noted by one child ("the picture is just the word"). One child expressed a design idea to turn spelling correction into a game ("you could guess a letter and it would tell if you were right or wrong"). Such a system could be more effective at teaching children how to spell while correcting their spelling.

CHAPTER 7

CONCLUSIONS

In this thesis, I explore solutions and document problems existing with spellchecking for children. Based on knowledge of children's spelling habits, we theorized the use of a phonetic encoding to find suitable spelling suggestions for children's spelling errors. This method greatly outperformed state-of-the-art methods while maintaining comparable efficiency. This method proved especially impressive when generating a relatively few amount of candidates which could quickly and reliably provide the intended word making the task of ranking easier. The unique application of lambdaMART LTR for spelling suggestions outperformed other machine learning methods and significantly improved the ranking of the generated spelling candidates. KidSpellλ's improvement over state-of-the-art methods was significant regardless of the context the spelling errors were made in, the grade level, spelling level, or native language of the user. Analysis of the relative feature importance from these models reinforce the importance of the KidSpell phonetic algorithm. We also show the importance of other features when addressing a child audience such as age of acquisition and word frequency. Features commonly used to assist with adult spellchecking, such as Levenshtein distance and n-gram scores, proved to be less valuable.

I further documented problems and addressed them at the user interface level. Modalities were identified to assist children in their selection of spelling suggestions.

The study of those audio and visual cues for a spellchecking interface demonstrate their importance on improving children's effective use of a spellchecker with conditions using the audio only cues showing the most promise. Those results lead to documented gaps and the design of an interface involving children as design partners. This gave us insights on how to improve spellchecking interfaces going forward that encourage children to address spelling errors.

7.1 Limitations and Future Work

While we made great progress on spelling errors made from both hand-written essays and typed search queries, there is a noticeable difference between the two. Our focus was on improving search and typed search queries proved to be the most difficult to correct. In this instance, hand-written essays errors are not a perfect proxy for the type of errors children can make while making search queries. More training data could help with handling typing errors. Incorporating boundary error detection could resolve some errors more commonly made while typing.

Further work remains when it comes to spelling error detection which was not addressed in this work. For example some spelling errors in our dataset are meant to be pop-culture terms (e.g., *optimus prime* or *roblox*). Since these words are not in our static dictionary, it is not possible to correct them with our current method.

Further examination on the effect of image cues could be explored that limits images to those that have concrete connections to words or curating the images that are used. When it comes to a good spellchecking interface, work remains to investigate how we can best teach children how to spell rather than what can quickly fix errors.

REFERENCES

[1] James F. Allen. *Natural Language Processing*, page 1218–1222. John Wiley and Sons Ltd., GBR, 2003.

[2] Oghenemaro Anuyah, Jerry Alan Fails, and Maria Soledad Pera. Investigating query formulation assistance for children. In *Proceedings of the 17th ACM Conference on Interaction Design and Children*, pages 581–586, 2018.

[3] GNU Aspell. Gnu aspell, Accessed 2020.

[4] Ion Madrazo Azpiazu, Nevena Dragovic, Maria Soledad Pera, and Jerry Alan Fails. Online searching and learning: Yum and other search tools for children and teachers. *Information Retrieval Journal*, 20(5):524–545, 2017.

[5] Donald R Bear, Marcia Invernizzi, Francine Johnston, and Shane Templeton. *Words their way: Word study for phonics, vocabulary, and spelling*. Merrill, 1996.

[6] Paul Bloom. *How children learn the meanings of words*. MIT press, 2002.

[7] Eric Brill and Robert C Moore. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th annual meeting on association for computational linguistics*, pages 286–293. Association for Computational Linguistics, 2000.

[8] Christopher JC Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81, 2010.

[9] Kenneth W Church and William A Gale. Probability scoring for spelling correction. *Statistics and Computing*, 1(2):93–103, 1991.

[10] W Bruce Croft, Donald Metzler, and Trevor Strohman. *Search engines: Information retrieval in practice*, volume 520. Addison-Wesley Reading, 2010.

[11] Fred J Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[12] Renato Cordeiro De Amorim and Marcos Zampieri. Effective spell checking methods using clustering algorithms. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 172–178, 2013.

[13] Sebastian Deorowicz and Marcin G Ciura. Correcting spelling errors by modelling their causes. *International journal of applied mathematics and computer science*, 15:275–285, 2005.

[14] Brody Downs, Oghenemaro Anuyah, Aprajita Shukla, Jerry Alan Fails, Maria Soledad Pera, , Katherine Landau Wright, and Casey Kennington. Kidspell: A child-oriented, rule-based, phonetic spellchecker. In *LREC*, page submitted. ACL, 2020.

[15] Brody Downs, Aprajita Shukla, Mikey Krentz, Maria Soledad Pera, Katherine Landau Wright, Casey Kennington, and Jerry Fails. Guiding the selection of child spellchecker suggestions using audio and visual cues. In *Proceedings of the Interaction Design and Children Conference*, pages 398–408, 2020.

[16] Nevena Dragovic, Ion Madrazo Azpiazu, and Maria Soledad Pera. " is sven seven?" a search intent module for children. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 885–888, 2016.

[17] Allison Druin, Elizabeth Foss, Hilary Hutchinson, Evan Golub, and Leshell Hatley. Children's roles using keyword search interfaces at home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 413–422, 2010.

[18] Sergio Duarte Torres, Djoerd Hiemstra, and Theo Huibers. Vertical selection in the information domain of children. In *Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries*, pages 57–66, 2013.

[19] Jens Edlund, Fredrik Edelstam, and Joakim Gustafson. Human pause and resume behaviours for unobtrusive humanlike in-car spoken dialogue systems. In *Proceedings of the EACL 2014 Workshop on Dialogue in Motion*, pages 73–77, 2014.

[20] Jerry Alan Fails, Mona Leigh Guha, Allison Druin, et al. Methods and techniques for involving children in the design of new technology for children. *Foundations and Trends® in Human–Computer Interaction*, 6(2):85–166, 2013.

[21] Jerry Alan Fails, Maria Soledad Pera, Oghenemaro Anuyah, Casey Kennington, Katherine Landau Wright, and William Bigirimana. Query formulation assistance for kids: What is available, when to help & what kids want. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, pages 109–120, 2019.

[22] VV Fomin and I Yu Bondarenko. A study of machine learning algorithms applied to gis queries spelling correction. *Komp'juternaja Lingvistika i Intellektual'nye Tehnologii*, 2018(17):185–199, 2018.

[23] Elizabeth Foss, Allison Druin, Robin Brewer, Phillip Lo, Luis Sanchez, Evan Golub, and Hilary Hutchinson. Children's search roles at home: Implications for designers, researchers, educators, and parents. *Journal of the American Society for Information Science and Technology*, 63(3):558–573, 2012.

[24] TN Gadd. Phonix: The algorithm. *Program*, 1990.

[25] Yasser Ganjisaffar, Andrea Zilio, Sara Javanmardi, Inci Cetindil, Manik Sikka, Sandeep Katumalla, Narges Khatib, Chen Li, and Cristina Lopes. qspell: Spelling correction of web search queries using ranking models and iterative correction. In *Spelling Alteration for Web Search Workshop*, page 15, 2011.

[26] Robert Garfinkel, Elena Fernandez, and Ram Gopal. Design of an interactive spell checker: optimizing the list of offered words. *Decision support systems*, 35(3):385–397, 2003.

[27] Joshua T. Goodman. A bit of progress in language modeling. *Computer Speech & Language*, 15(4):403 – 434, 2001.

[28] Tatiana Gossen. Large-scale analysis of children's queries and search interactions. In *Search Engines for Children*, pages 79–85. Springer, 2015.

[29] Tatiana Gossen, Thomas Low, and Andreas Nürnberger. What are the real differences of children's and adults' web search. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1115–1116, 2011.

[30] Tatiana Gossen, Marcus Nitsche, and Andreas Nürnberger. Knowledge journey: A web search interface for young users. In *Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval*, pages 1–10, 2012.

[31] Daphne Greenberg, Linnea C Ehri, and Dolores Perin. Do adult literacy students make the same word-reading and spelling errors as children matched for word-reading age? *Scientific Studies of Reading*, 6(3):221–243, 2002.

[32] Mona Leigh Guha, Allison Druin, and Jerry Alan Fails. Cooperative inquiry revisited: Reflections of the past and guidelines for the future of intergenerational co-design. *International Journal of Child-Computer Interaction*, 1(1):14–23, 2013.

[33] Jacek Gwizdka and Dania Bilal. Analysis of children's queries and click behavior on ranked results and their thought processes in google search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval*, pages 377–380, 2017.

[34] Matthias Hagen, Martin Potthast, Marcel Gohsen, Anja Rathgeber, and Benno Stein. A large-scale query spelling correction corpus. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1261–1264, 2017.

[35] Juan Pablo Hourcade. Child-computer interaction. *Self, Iowa City, Iowa*, 2015.

[36] Yinghao Huang, Yi Lu Murphey, and Yao Ge. Automotive diagnosis typo correction using domain knowledge and machine learning. In *2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 267–274. IEEE, 2013.

[37] R Malatesha Joshi, Rebecca Treiman, Suzanne Carreker, and Louisa C Moats. How words cast their spell. *American Educator*, 32(4):6–16, 2008.

[38] Yvonne Kammerer and Maja Bohnacker. Children's web search with google: the effectiveness of natural language queries. In *proceedings of the 11th international conference on interaction design and children*, pages 184–187, 2012.

[39] Alex Kuhn, Clara Cahill, Chris Quintana, and Shannon Schmoll. Using tags to encourage reflection and annotation on data during nomadic inquiry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 667–670, 2011.

[40] Victor Kuperman, Hans Stadthagen-Gonzalez, and Marc Brysbaert. Age-of-acquisition ratings for 30,000 english words. *Behavior research methods*, 44(4):978–990, 2012.

[41] Adrey Kutuzov and Elizaveta Kuzmenko. Semi-automated typical error annotation for learner english essays: integrating frameworks. In *Proceedings of the 4th workshop on NLP for Computer Assisted Language Learning at NODALIDA 2015, Vilnius, 11th May, 2015*, number 114 in 1, pages 35–41. Linköping University Electronic Press, 2015.

[42] Monica Landoni, Davide Matteri, Emiliana Murgia, Theo Huibers, and Maria Soledad Pera. Sonny, cerca! evaluating the impact of using a vocal assistant to search at school. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 101–113. Springer, 2019.

[43] Jim Lawley. Spelling: computerised feedback for self-correction. *Computer Assisted Language Learning*, 29(5):868–880, 2016.

[44] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[45] Yanen Li, Huizhong Duan, and ChengXiang Zhai. Cloudspeller: Spelling correction for search queries by using a unified hidden markov model with web-scale resources. In *Spelling Alteration for Web Search Workshop*, pages 10–14. Citeseer, 2011.

[46] Ion Madrazo Azpiazu, Nevena Dragovic, Oghenemaro Anuyah, and Maria Soledad Pera. Looking for the movie seven or sven from the movie frozen? a multi-perspective strategy for recommending queries for children. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval*, pages 92–101, 2018.

[47] Joseph E Michaelis and Bilge Mutlu. Supporting interest in science learning with a social robot. In *Proceedings of the 18th ACM International Conference on Interaction Design and Children*, pages 71–82, 2019.

[48] Roger Mitton. Ordering the suggestions of a spellchecker without using context. *Natural Language Engineering*, 15(2):173–192, 2009.

[49] Harshit Pande. Effective search space reduction for spell correction using character neural embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 170–174, 2017.

[50] Lawrence Philips. Hanging on the metaphone. *Computer Language*, 7(12):39–43, 1990.

[51] Mark Sadoski, Ernest T Goetz, and Joyce B Fritz. A causal model of sentence recall: Effects of familiarity, concreteness, comprehensibility, and interestingness. *Journal of reading behavior*, 25(1):5–16, 1993.

[52] Rodrygo LT Santos, Craig Macdonald, and Iadh Ounis. Learning to rank query suggestions for adhoc and diversity search. *Information Retrieval*, 16(4):429–451, 2013.

[53] RJW Sluis, Ivo Weevers, CHGJ Van Schijndel, Lyuba Kolos-Mazuryk, Siska Fitrianie, and JBOS Martens. Read-it: five-to-seven-year-old children learn to read in a tabletop environment. In *Proceedings of the 2004 conference on Interaction design and children: building a community*, pages 73–80, 2004.

[54] Thanh Vu, Alistair Willis, Udo Kruschwitz, and Dawei Song. Personalised query suggestion for intranet search with temporal user profiling. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, pages 265–268, 2017.

[55] Chao Wang and Rongkai Zhao. Multi-candidate ranking algorithm based spell correction. In *In The 2019 SIGIR Workshop On eCommerce, Paris, France*, 2019.

[56] Casey Whitelaw, Ben Hutchinson, Grace Y Chung, and Gerard Ellis. Using the web for language independent spellchecking and autocorrection. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 890–899. Association for Computational Linguistics, 2009.

[57] Emmanuel J Yannakoudakis and David Fawthrop. An intelligent spelling error corrector. *Information Processing & Management*, 19(2):101–108, 1983.

APPENDIX A

MACHINE LEARNING EVALUATIONS

The figures in Figure A.1 and Figure A.2 describe the hit-rate and MRR for the machine learning models explored as described in Section 4.2. The model labeled as *lambdaMART* is the same as KidSpellλ.

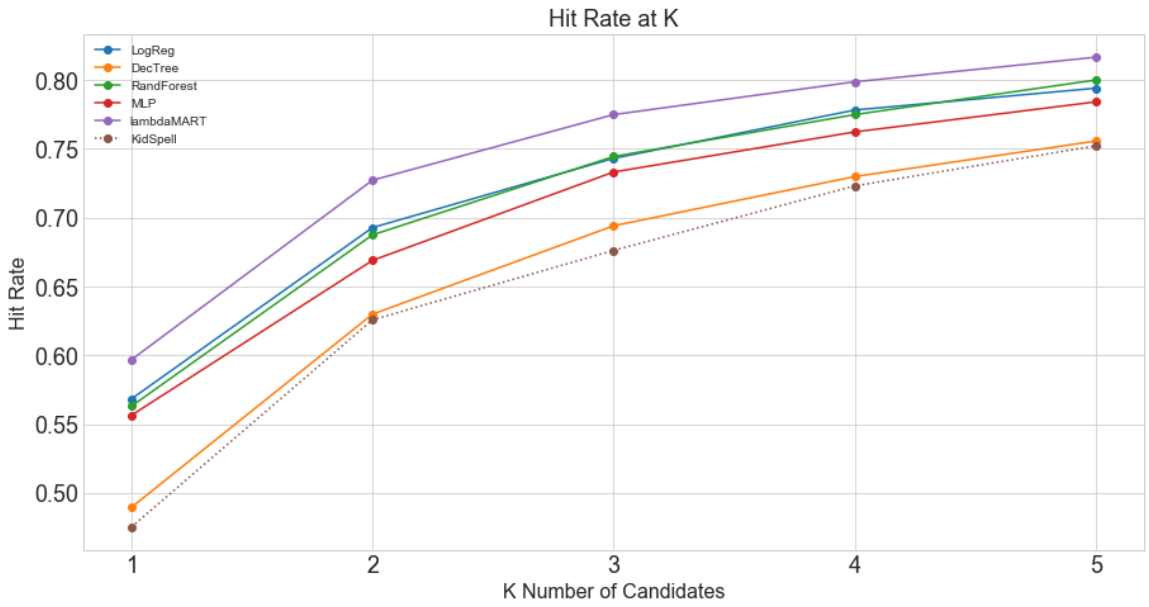

Figure A.1: Hit-Rate for various k (number of suggestions) on spelling errors made in typed search queries with a comparison between different machine learning models.

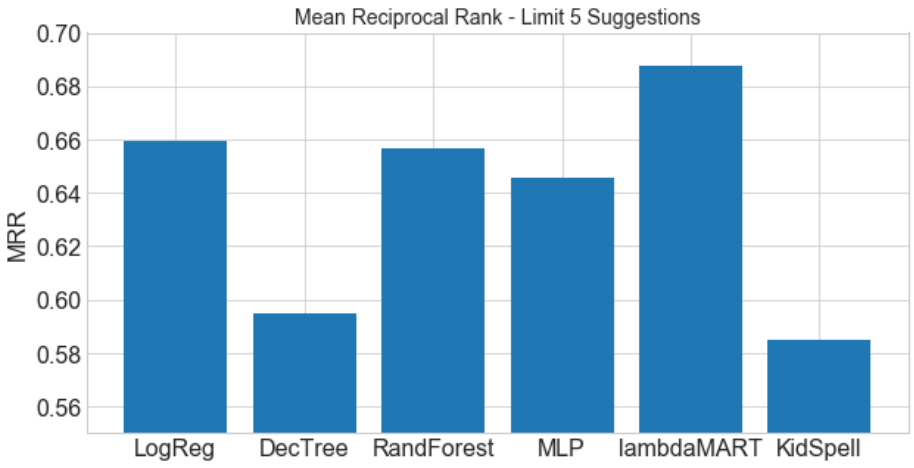

Figure A.2: MRR using top 5 suggestions for typed search queries with a comparison between different machine learning models.