Abstract: Automatic spelling correction has been receiving sustained research attention. Although each article contains a brief introduction to the topic, there is a lack of work that would summarize the theoretical framework and provide an overview of the approaches developed so far. Our survey selected papers about spelling correction indexed in Scopus and Web of Science from 1991 to 2019. The first group uses a set of rules designed in advance. The second group uses an additional model of context. The third group of automatic spelling correction systems in the survey can adapt its model to the given problem. The summary tables show the application area, language, string metrics, and context model for each system. The survey describes selected approaches in a common theoretical framework based on Shannon’s noisy channel. A separate section describes evaluation methods and benchmarks.

Keywords: spelling correction; natural language processing; diacritization; error model; context model

1. Introduction

There are many possible ways to write the same thing. Written text sometimes looks different from what the reader or the author expects. Creating apprehensive and clear text is not a matter of course, especially for people with a different mother language. An unusually written word in a sentence makes a spelling error.

A spelling error makes the text harder to read and, worse, harder to process. Natural language processing requires normalized forms of a word because incorrect spelling or digitization of text decreases informational value. A spelling error, for example, in a database of medical records, diminishes efficiency of the diagnosis process, and incorrectly digitized archive documents can influence research or organizational processes.

A writer might not have enough time or ability to correct spelling errors. Automatic spelling correction (ASC) systems help to find the intended form of a word. They identify problematic words and propose a set of replacement candidates. The candidates are usually sorted according to their expected fitness with the spelling error and the surrounding context. The best correction can be selected interactively or automatically.

Interactive spelling correction systems underline incorrectly written words and suggest corrections. A user of the system selects the most suitable correction. This scenario is common in computer-assisted proofreading that helps with the identification and correction of spelling errors. Interactive spelling correction systems improve the productivity of professionals working with texts, increase convenience when using mobile devices, or correct Internet search queries. They support learning a language, text input in mobile devices, and web search engines. Also, interactive spelling correction systems are a component of text editors and office systems, optical character recognition (OCR) systems, and databases of scanned texts.
Most current search engines can detect misspelled search queries. The suggestion is shown interactively for each given string prefix. A recent work by Cai and de Rijke [1] reviewed approaches for correcting search queries.

A large quantity of text in databases brought new challenges. An automatic spelling correction system can be a part of a natural language processing system. Text in the database has to be automatically corrected because interactive correction would be too expensive. The spelling correction system automatically selects a correction candidate according to the previous and following texts. Noninteractive text normalization can improve the performance of information retrieval or semantic analysis of a text.

Figure 1 displays the process of correction-candidate generation and correction. The error and context models contribute to ranking of the candidate words. The result of automatic correction is a sequence of correction candidates with the best ranking.

![Figure 1. Interactive processes of error production and correction.](image)

In the next section, you’ll find an explanation of the method we used to select and sort the articles in this report. Subsequently, in Section 3, we describe the characteristic spelling errors and divide them into groups according to how they originated. Section 4 defines the task of correcting spelling errors and describes the ASC system. This survey divides the ASC systems into three groups, each with its section: a priori spelling correction (Section 5), spelling correction in the context (Section 6), and spelling correction with a learning error model (Section 7). Section 8 introduces the methods of evaluation and benchmarking. The concluding Section 9 summarizes the survey and outlines trends in the research.

2. Methodology


First, we searched the indices with a search query “spelling correction” for the years 1991–2019. Scopus returned 1315 results, WoS returned 794 results. We excluded 149 errata, 779 corrections, 7 editorials, 45 reviews, and around 140 papers without any citations from both collections. We removed 250 duplicates, and we received 740 results (440 journal articles and 300 conference papers). We read the titles and abstracts of the remaining papers and removed 386 works that are not relevant to the topic of automatic spelling correction.

We examined the remaining 354 documents. Then, we removed articles without clear scientific contribution to spelling correction, without proper evaluation, or that just repeated already known things. We examined, sorted, and put the remaining 119 items into tables. We included additional references that explain essential theoretical concepts and survey papers about particular topics in the surrounding text.
First, we defined the spelling correction problem and established a common theoretical framework. We described the three main components of a spelling correction system.

This work divides the selected papers into three groups. The first group uses a set of expert rules to correct a spelling error. The second group adds a context model to rearrange the correction candidates with the context. The third group learns error patterns from a training corpus.

Each group of methods has its own section with a summarizing table. The main part of the survey is the summary tables. The tables briefly describe the application area, language, error model, and context model of the spelling correction systems. The tables are accompanied by a description of the selected approaches.

The rows in the tables are sorted chronologically and according to author. We selected chronological order because it shows the general scientific progress in spelling correction in the particular components of the spelling correction system. An additional reference in the table indicates if one approach enhances the previous one.

Special attention is paid to the evaluation methods. This section identifies the most frequent evaluation methods, benchmarks and corpora.

3. Spelling Errors

The design of an automatic spelling correction system requires knowledge of the creation process of a spelling error [3]. There are several works about spelling errors. A book by Mitton [4] analyzed spelling-error types and described approaches to construct an automatic spelling correction system. The authors in Yannakoudakis and Fawthrop [5] demonstrated that the clear majority of spelling errors follow specific rules on the basis of phonological and sequential considerations. The paper [5] introduced and described three categories of spelling errors (consonantal, vowel, and sequential) and presented the analysis results of 1377 spelling error forms.

Moreover, the authors in Kukich [2], Toutanova and Moore [6], and Pirinen and Lindén [7] divided spelling errors into two categories according to their cause:

1. Cognitive errors (also called orthographic or consistent): They are caused by the disabilities of the person that writes the text. The correct way of writing may be unknown to the writer. The writer could have dyslexia, dysgraphia, or other cognitive problems. The person writing the text could just be learning the language and not know the correct spelling. This set of errors is language- and user-specific because it is more dependent on using the rules of the language [7].

2. Typographic errors (also called conventional): They are usually related to technical restrictions of the input device (physical or virtual keyboard, or OCR system) or depend on the conditions of the environment. Typing in haste often causes substitution of two close keys. Typographic errors caused by hasty typing are usually language-agnostic (unrelated to the language of the writer), although they can depend on local keyboard mapping or a localized OCR system [7].

Examples of typographic and cognitive spelling errors are in Table 1.

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cognitive error:</td>
<td>I don’t know the correct spelling of Levenshtein distance.</td>
</tr>
<tr>
<td>Typographic:</td>
<td>This sentence was typed in haser.</td>
</tr>
<tr>
<td>Typographic (OCR):</td>
<td>SUPPLEMENTALy INFORMATION.</td>
</tr>
<tr>
<td>Typographic (Diacritic):</td>
<td>The authors of this article are Daniel Hladek, Matus Pleva and Jan Stas.</td>
</tr>
</tbody>
</table>

Note: The spelling errors are underlined.

OCR errors are a particular type of typographic error caused by software. The process of document digitization and optical character recognition often omits or replaces some letters in a typical way. Spelling correction is part of postprocessing of the digitized document because OCR systems are
usually proprietary and difficult to adapt. Typical error patterns appear in OCR texts [8]. The standard set for evaluation of an OCR spelling correction system is the TREC-5 Confusion Track [9].

Some writing systems (such as Arabic, Vietnamese, or Slovak) use different character variants that change the meaning of the word. The authors in [10] confirmed that the omission of diacritics is a common type of spelling error in Brazilian Portuguese. Texts in Modern Standard Arabic are typically written without diacritical markings [11]. This is a typographic error when the author omits additional character markings and expects the reader to guess the original meaning. The missing marks usually present short vowels or modification of the letter. They are placed either above or below the graphemes. The process of adding vowels and other diacritic marks to Arabic text can be called diacritization or vowelization [11]. Azmi and Almajed [12] focused on the problem of Arabic diacritization (adding missing diacritical markings to Arabic letters) and proposed an evaluation metric, and Asahiah et al. [13] published a survey of Arabic diacritization techniques.

4. Automatic Spelling Correction

An automatic spelling correction system detects a spelling error and proposes a set of candidates for correction (see Figure 2). Kukich [2] and Pirinen and Lindén [7] divide the whole process into three steps:

1. detection of an error;
2. generation of correction candidates;
3. ranking of candidate corrections.

Figure 2. Process of automatic spelling correction.

4.1. Error Detection

A word could either be new or just uncommon, could be a less-known proper name, or could belong to another language. However, a correctly spelled word could be semantically incorrect in a sentence. Kukich [2] divided spelling errors according to the dictionary of correct words:

- real-word errors, where the word is spelled incorrectly but its form is in the dictionary of correct words, and
- non-word errors, where the incorrect word form is not in the dictionary of correct words.

Most spelling correction systems detect a non-word error by searching for it in a dictionary of correct words. This step requires a fast-lookup method such as hash table [14] or search tree [15,16].

Many non-word error spelling correction systems use open-source a priori spelling systems, such as Aspell or Hunspell for error detection, correction-candidate generation, and preliminary candidate ranking.

An automatic spelling correction system identifies real-word errors by semantic analysis of the surrounding context. More complex error-detection systems may be used to detect words that are correctly spelled but do not fit into the syntactic or semantic context. Pirinen and Lindén [7] called it real-word error detection in context.
Real-word errors are hard to detect because detection requires semantic analysis of the context. The authors in [17] used a language model to detect and correct a homophonic real-word error in the Bangla language. The language model identifies words that are improbable with the current context.

Boytsov [18] examined methods for indexing a dictionary with approximate matching. Deorowicz and Ciura [19] claim that a lexicon of all correct words could be too large. Too large a lexicon can lead to many real-word errors or misdetection of obscure spellings.

The situation is different for languages where words are not separated by spaces (for example, Chinese). The authors in [20] transformed characters into a fixed-dimensional word-vector space and detected spelling errors by conditional random field classification.

4.2. Candidate Generation

ASC systems usually select correction candidates from a dictionary of correct words after detection of a spelling error. Although it is possible to select all correct words as correction candidates, it is reasonable to restrict the search space and to inspect only words that are similar to the identified spelling error.

Zhang and Zhang [21] stated that the task of similarity joining is to find all pairs of strings for which similarities are above a predetermined threshold, where the similarity of two strings is measured by a specific distance function. Kernighan et al. [22] proposed a simplification to restrict the candidate list to words that differ with just one edit operation of the Damerau–Levenshtein edit distance—substitution, insertion, deletion, or replacement of succeeding letters [23].

The spelling dictionary generates correction candidates for the incorrect word by approximately searching for similar words. The authors in [24] used a character-level language model trained on a dictionary of correct words to generate a candidate list. Reffle [25] used a Levenshtein automaton to propose the correction candidates. Methods of approximate searching were outlined in a survey published by Yu et al. [26].

An index often speeds up an approximate search in the dictionary. The authors in [19,27] converted the lexicon into a finite-state automaton to speed up searching for a similar string.

4.3. Ranking Correction Candidates

A noisy-channel model proposed by Shannon [28] described the probabilistic process of producing an error. The noisy channel transfers and distorts words (Figure 3).

\[
\text{Intended Word} \xrightarrow{\text{Noise}} \text{Noisy Channel} \xrightarrow{\text{Distorted Word}}
\]

\text{Figure 3. Word distorted by noisy channel.}

The noisy-channel model expresses similarity between two strings as a probability of transforming one string into another. Probability \( P(s|w) \) that a string \( s \) is produced instead of word \( w \) describes how similar the two strings are. The similarity between two strings is defined by an expert or depends on a training corpus with error patterns.

A more formal definition of automatic spelling correction uses the maximum-likelihood principle. Brill and Moore [29] defined the automatic spelling correction of a possibly incorrect word \( s \) as finding the best correction candidate \( w_b \) from a list of possible correction candidates \( w_i \in W \) with the highest un-normalized probability:

\[
w_b = \arg \max_{w_i \in C(s)} P(s|w_i)P(w_i),
\]
where $P(s|w_i)$ is the probability of producing string $s$ instead of word $w_i$ and $P(w_i)$ is the probability of producing word $w_i$. $C(s)$ is a function that returns valid words from dictionary $W$ that serve as correction candidates for erroneous string $s$.

4.4. Components of Automatic Spelling Correction Systems

Equation (1) by Brill and Moore [29] identified three components of an automatic spelling correction system. The components are depicted in Figure 4:

1. Dictionary: It detects spelling errors and proposes correction candidates $w_i \in C$ for each input token. $C(s)$ is a list of correction candidates $w_i$ for a given token $s$. The list of correction candidates belongs to the set of all correct words ($C(s) \in W$). If the dictionary does not propose any candidate, the word is considered correct.

2. Error model (channel model) $P(s|w_i)$: It is an essential component of the automatic spelling correction system. It measures the "fitness" of the correction candidate with the corrected string. The model expresses the similarity of strings $w_i$ and $s$ or the probability of producing string $s$ instead string $w_i$. This measure does not have to be purely probabilistic but can be similar to a distance between the two strings. The non-probabilistic string distance can always be converted into probabilistic string similarity (see Equation (3) in Section 6). An error model allows for identification of the most probable errors and consequently the most probable original forms.

3. Context model (source model $P(w_i)$, the prior model of word probabilities): This expresses the probability of correct word occurrence and often takes the context of the word into account. Candidates that best fit into the current context have a higher probability of being the intended word. The context model focuses on finding the best correction candidate by using the context of the incorrect word and statistical methods of classification. The model observes features that are outside the inspected word and improves the evaluation of candidate words. It can detect an unusual sequence of features and identify real-word errors.

5. Spelling Correction with a Priori Error Model

A combination of error and context models is often not necessary. In some scenarios, a set of predefined transcription rules can correct a spelling error. An expert identifies characteristic string transcriptions. These rules are given in advance (a priori) by someone who understands the problem.

Approaches in this group detect non-word errors and propose a list of correction candidates that are similar to the original word (presented in Table 2). The a priori error model works as a guide in the search for the best-matching original word; best-matching words are proposed first, and it is easy to select the correction.

A schematic diagram for an ASC system with a priori error model is in Figure 5. The input of the a priori error model is an erroneous word. The spelling system applies one or several transcription operations to the spelling error to create a correction candidate. The rank of the correction candidate depends on the weights of the transcription rules. The output of the a priori error model is a sorted list with correction candidates.
Table 2. Summary of a priori spelling correction systems.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Application</th>
<th>Language</th>
<th>Error Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Khairul Islam et al. [30], 2019</td>
<td>General</td>
<td>Bangla</td>
<td>LD</td>
</tr>
<tr>
<td>Haswezi et al. [31], 2019</td>
<td>General</td>
<td>Kurdish</td>
<td>LD, DLD, LCS</td>
</tr>
<tr>
<td>Thatiprayoon et al. [32], 2018</td>
<td>Search query</td>
<td>Thai</td>
<td>LD, Soundex</td>
</tr>
<tr>
<td>Christanti et al. [33], 2018</td>
<td>General</td>
<td>Indonesian</td>
<td>DLD</td>
</tr>
<tr>
<td>Hagen et al. [34], 2017</td>
<td>Search query</td>
<td>English</td>
<td>DLD</td>
</tr>
<tr>
<td>Sakuntharaj and Mahesan [35], 2016</td>
<td>General</td>
<td>Tamil</td>
<td>LD, common n-grams</td>
</tr>
<tr>
<td>Vohl et al. [36], 2014</td>
<td>OCR, historical</td>
<td>Old German</td>
<td>Interactive</td>
</tr>
<tr>
<td>Rees [37], 2014</td>
<td>Animal taxonomy</td>
<td>Latin</td>
<td>Soundex</td>
</tr>
<tr>
<td>Mühlberger et al. [38], 2014</td>
<td>OCR, historical</td>
<td>German</td>
<td>Interactive</td>
</tr>
<tr>
<td>Patrick and Nguyen [39], 2014</td>
<td>General, medical</td>
<td>English</td>
<td>Interactive</td>
</tr>
<tr>
<td>Kashefi et al. [40], 2013</td>
<td>Diacritization</td>
<td>Farsi</td>
<td>Modified DLD</td>
</tr>
<tr>
<td>Andrade et al. [41], 2012</td>
<td>General</td>
<td>Portuguese</td>
<td>DLD</td>
</tr>
<tr>
<td>Sha et al. [42], 2011</td>
<td>General</td>
<td>Chinese</td>
<td>Keyboard-based edit distance</td>
</tr>
<tr>
<td>Refle [25], 2011</td>
<td>OCR, historical</td>
<td>Old German</td>
<td>LD, FSA</td>
</tr>
<tr>
<td>Naji and Savoy [43], 2011</td>
<td>General, historical</td>
<td>Middle High German</td>
<td>Stemmer</td>
</tr>
<tr>
<td>Deorowicz and Ciura [39], 2005</td>
<td>General</td>
<td>English</td>
<td>FSA</td>
</tr>
<tr>
<td>UzzZaman and Khan [45], 2005</td>
<td>General</td>
<td>Bangla</td>
<td>Bangla double metaphor</td>
</tr>
<tr>
<td>Vilares et al. [27], 2004</td>
<td>General</td>
<td>Galician</td>
<td>FSA</td>
</tr>
<tr>
<td>van Delden et al. [46], 2004</td>
<td>General</td>
<td>English</td>
<td>LD, stemming</td>
</tr>
<tr>
<td>Schulz and Mihov [47], 2002</td>
<td>General</td>
<td>Bulgarian</td>
<td>FSA</td>
</tr>
<tr>
<td>Taghva and Stofsky [48], 2001</td>
<td>OCR</td>
<td>English</td>
<td>Interactive + LCS subsequence</td>
</tr>
<tr>
<td>Vagelatos et al. [49], 1995</td>
<td>General</td>
<td>Greek</td>
<td>Interactive</td>
</tr>
</tbody>
</table>

Note: DLD, Damerau–Levenshtein distance; FSA, finite-state automaton; LCS, longest common subsequence; LD, Levenshtein distance; OCR, optical character recognition.

Figure 5. A priori spelling correction.

The most commonly used open-source spelling systems are Aspell (http://aspell.net) and Hunspell (http://hunspell.github.io/). Hunspell is a variant of Aspell with a less restrictive license, used in LibreOffice word processor, Firefox web browser, and other programs. They are available as a standalone text filter or as a compiled component in other spelling systems or programs. The basic component of the Aspell system is a dictionary of correct words, available for many languages. The dictionary file contains valid morphological units for the given language (prefixes, suffixes, or stems). The dictionary is compiled into a state machine to speed up searching for correction candidate words.

Aspell searches for sounds-like equivalents (computed for English words by using the Metaphone algorithm) up to a given edit distance (the Damerau–Levenshtein distance) [50]. The detailed operation of the spelling correction of Aspell is described in the manual (http://aspell.net/man-html/Aspell-Suggestion-Strategy.html#Aspell-Suggestion-Strategy).

5.1. Edit Distance

Edit distance expresses the difference between two strings as a nonnegative real number by counting edit operations that are required to transform one string into another. The two most commonly used edit distances are the Levenshtein edit distance [51] and the Damerau–Levenshtein distance [52]. Levenshtein identifies atomic edit operations such as

- Substitution: replaces one symbol into another;
- Deletion: removes a symbol (or replaces it with an empty string $\epsilon$); and
- Insertion: adds a symbol or replaces an empty string $\epsilon$ with a symbol.

In addition, the Damerau–Levenshtein distance adds the operation of

- Transposition, which exchanges two subsequent symbols.
The significant difference between the Levenshtein distance (LD) and the Damerau–Levenshtein distance (DLD) is that the Levenshtein distance does not consider letter transposition. The edit operation set proposed by Levenshtein [51] did not consider transposition as an edit operation because the transposition of two subsequent letters can be substituted by deletion and insertion or by two substitutions. The Levenshtein distance allows for representation of the weights of edit operations by a single letter-confusion matrix, which is not possible for DLD distance.

Another variation of edit distance is longest common subsequence (LCS) [53]. It considers only insertion and deletion edit operations. The authors in [54] proposed an algorithm for searching for the longest common sub-string with the given number of permitted mismatches. More information about longest-common-subsequence algorithms can be found in a survey [55].

5.2. Phonetic Algorithms

Many languages have difficult rules for pronunciation and writing, and it is very easy to make a spelling mistake if rules for writing a certain word are not familiar to the writer. A word is often replaced with a similarly sounding equivalent with a different spelling.

An edit operation in the phonetic algorithm describes how words are pronounced. They recursively replace phonetically important parts of a string into a special representation. If the phonetic representation of two strings is equal, the strings are considered equal. In other words, a phonetic algorithm is a binary relation of two strings that tells whether two strings are pronounced in a similar way:

$$D(s_s, s_t) \rightarrow 0 \text{ or } 1 .$$

(2)

The phonetic algorithm is able to identify a group of phonetically similar words to some given string (e.g., to some unknown proper noun). It helps to identify names that are pronounced in a similar way or to discover the original spelling of an incorrectly spelled word. Two strings are phonetically similar only if their phonetic forms are equal.

Phonetic algorithms for spelling corrections and record linkage are different from phonetic algorithms used for speech recognition because they return just an approximation of the true phonetic representation.

One of the first phonetic algorithms is Soundex (U.S. Patent US1435663). Its original purpose was the identification of similar names for the U.S. Census. The algorithm transforms a surname or name so that names with a similar pronunciation have the same representation. It allows for the identification of similar or possibly the same names. The most phonetically important letters are consonants. Most vowels are dropped (except for in the beginning), and similar consonants are transformed into the same representation. Other phonetic algorithms are Shapex [56] and Metaphone [57]. Evaluation of several phonetic-similarity algorithms on the task of cognate identification was done by Kondrak and Sherif [58].

6. Spelling Correction in Context

An a priori model is often not sufficient to find out the best correction because it takes only incorrect word into account. The spelling system would perform better if it could distinguish whether the proposed word fits with its context. It is hard to decide which correction is more useful if we do not know the surrounding sentence. For example, if a correction for string “smilly” is “smelly”, the correction “smiley” can be more suitable for some contexts.

Approaches in this group are summarized in Tables 3 and 4. The components and their functions are displayed in Figure 4. The authors in [59] described multiple methods of correction with context. This group of automatic spelling correction systems use a probabilistic framework by Brill and Moore [29] defined in the Equation (1). The error models in this group usually use the a priori rules (edit distance and phonetic algorithms). The context model is usually an $n$-gram language model. Some approaches noted below use a combination of multiple statistical models.
Table 3. Spelling correction systems with learning of context model—part I.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Application</th>
<th>Language</th>
<th>Context Model</th>
<th>Error Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azmi et al. [60], 2019</td>
<td>General, OCR</td>
<td>Arabic</td>
<td>LM</td>
<td>LD, DLD</td>
</tr>
<tr>
<td>Dong et al. [61], 2019</td>
<td>MT</td>
<td>Uygur, Chinese</td>
<td>LM, BLEU score</td>
<td>LD</td>
</tr>
<tr>
<td>Yazdani et al. [62], 2019</td>
<td>Medical</td>
<td>Farsi</td>
<td>LM</td>
<td>DLD</td>
</tr>
<tr>
<td>Damnati et al. [63], 2018</td>
<td>POS</td>
<td>French</td>
<td>Word embedding</td>
<td>DLD</td>
</tr>
<tr>
<td>Dashti [64], 2018</td>
<td>General</td>
<td>English</td>
<td>LM</td>
<td>CFG</td>
</tr>
<tr>
<td>Dong and Purwarianti [65], 2018</td>
<td>General</td>
<td>Indonesian</td>
<td>LM, POS, Viterbi</td>
<td>DLD</td>
</tr>
<tr>
<td>Heyman et al. [66], 2018</td>
<td>General</td>
<td>Dutch</td>
<td>Suffix probability</td>
<td>BiLSTM</td>
</tr>
<tr>
<td>Mashod Rana et al. [67], 2018</td>
<td>General</td>
<td>Bangla</td>
<td>Golding and Schabes [67]</td>
<td>WCS</td>
</tr>
<tr>
<td>Dziadek et al. [68], 2017</td>
<td>Medical ontology</td>
<td>Swedish</td>
<td>LM, POS</td>
<td>LD</td>
</tr>
<tr>
<td>Sorokin [69], 2017</td>
<td>General</td>
<td>Russian</td>
<td>LM, LR</td>
<td>LD, Metaphone</td>
</tr>
<tr>
<td>Zhao et al. [70], 2017</td>
<td>General</td>
<td>Chinese</td>
<td>CRF, decoder</td>
<td>Graph</td>
</tr>
<tr>
<td>de Mendonça Almeida et al. [71], 2016</td>
<td>General</td>
<td>Brazilian Portuguese</td>
<td>Decision tree</td>
<td>Modified Soundex</td>
</tr>
<tr>
<td>Lv et al. [72], 2016</td>
<td>OCR, Medical</td>
<td>Chinese</td>
<td>LM, ME</td>
<td>WCS</td>
</tr>
<tr>
<td>Melero et al. [73], 2016</td>
<td>General</td>
<td>Spanish</td>
<td>LM</td>
<td>WCS</td>
</tr>
<tr>
<td>Mirzababaei and Falli [74], 2016</td>
<td>General</td>
<td>Farsi, English</td>
<td>LM, SVM, PMI</td>
<td>DLD</td>
</tr>
<tr>
<td>Sorokin and Shavrina [75], 2016</td>
<td>General</td>
<td>Russian</td>
<td>LM, LR</td>
<td>LD</td>
</tr>
<tr>
<td>Vilares et al. [76], 2016</td>
<td>IR</td>
<td>Cross-language</td>
<td>POS</td>
<td>Character n-grams, DLD</td>
</tr>
<tr>
<td>Lhoussain et al. [77], 2015</td>
<td>General</td>
<td>Arabic</td>
<td>LM</td>
<td>Interactive</td>
</tr>
<tr>
<td>Ferrero et al. [78], 2014</td>
<td>General, proofreading</td>
<td>Spanish</td>
<td>Bayes</td>
<td>Letter n-grams, DLD</td>
</tr>
<tr>
<td>Miangah [14], 2014</td>
<td>General</td>
<td>Farsi</td>
<td>Word frequency</td>
<td></td>
</tr>
<tr>
<td>Pirinen and Lindén [7], 2014</td>
<td>General</td>
<td>Finish, Greenlandic</td>
<td>WFST, LM</td>
<td>WFST</td>
</tr>
<tr>
<td>Sagidiinos et al. [79], 2014</td>
<td>General</td>
<td>Greek</td>
<td>Id3, C4.5, k-NN, naïve Bayes, RF</td>
<td>Suffix</td>
</tr>
</tbody>
</table>

Note: BLEU, bilingual evaluation understudy; BiLSTM, bidirectional long short-term memory; CFG, context-free grammar; CRF, conditional random fields; IR, information retrieval; k-NN, k-nearest neighbors; LM, language model; LR, linear regression; ME, maximum entropy; POS, part-of-speech tagging; PMI, pointwise mutual information; RF, random forests; SVM, support vector machine; WCS, word-confusion set; WFST, weighted finite-state transducer.
Table 4. spelling correction systems with learning of context model—part II.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Application</th>
<th>Language</th>
<th>Context Model</th>
<th>Error Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ehsan and Faili [80], 2013</td>
<td>General</td>
<td>Farsi, English</td>
<td>SMT, ME</td>
<td>DLD</td>
</tr>
<tr>
<td>Hladek et al. [81], 2013</td>
<td>General</td>
<td>Slovak</td>
<td>LM, HMM</td>
<td>Aspell</td>
</tr>
<tr>
<td>Flor [59], 2012</td>
<td>General, proofreading</td>
<td>English</td>
<td>LM, Bouma [82]</td>
<td>Custom</td>
</tr>
<tr>
<td>Alkanhal et al. [83], 2012</td>
<td>General</td>
<td>Arabic</td>
<td>A-star, LM</td>
<td>DLD</td>
</tr>
<tr>
<td>Grozea [84], 2012</td>
<td>General</td>
<td>Romanian</td>
<td>LM, HMM</td>
<td>Trivial WCS</td>
</tr>
<tr>
<td>Stüker et al. [85], 2011</td>
<td>General, diagnosis</td>
<td>German</td>
<td>HMM, LM</td>
<td>Phonetic algorithm</td>
</tr>
<tr>
<td>Wong and Glance [86], 2011</td>
<td>General, medical</td>
<td>English</td>
<td>Bayes</td>
<td>Aspell</td>
</tr>
<tr>
<td>Abdulkader and Casey [87], 2009</td>
<td>OCR</td>
<td>English</td>
<td>ANN</td>
<td>Interactive</td>
</tr>
<tr>
<td>Ahmed et al. [50], 2009</td>
<td>Search query</td>
<td>English</td>
<td>Ternary search trees, letter n-grams</td>
<td>Trivial WCS</td>
</tr>
<tr>
<td>Farooq et al. [88], 2009</td>
<td>Handwritten OCR</td>
<td>English</td>
<td>Topic LM, ME</td>
<td>Trivial WCS</td>
</tr>
<tr>
<td>Carlson and Fette [89], 2007</td>
<td>General</td>
<td>English</td>
<td>Banko and Brill [90]</td>
<td>Aspell</td>
</tr>
<tr>
<td>Mykowiecka and Marciniak [91], 2006</td>
<td>General, medical</td>
<td>Polish</td>
<td>LM</td>
<td>Modified LD</td>
</tr>
<tr>
<td>Héja and Surján [92], 2003</td>
<td>General, medical</td>
<td>Hungarian</td>
<td>n-gram tree</td>
<td>Interactive</td>
</tr>
<tr>
<td>Jin et al. [93], 2003</td>
<td>OCR</td>
<td>English</td>
<td>ME</td>
<td>WCS</td>
</tr>
<tr>
<td>Ruch et al. [94], 2003</td>
<td>General, medical</td>
<td>English, French</td>
<td>POS, ME, WSD</td>
<td>Interactive</td>
</tr>
<tr>
<td>Li and Wang [95], 2002</td>
<td>General</td>
<td>Chinese</td>
<td>Golding and Roth [96]</td>
<td>LD</td>
</tr>
<tr>
<td>Banko and Brill [90], 2001</td>
<td>General</td>
<td>English</td>
<td>Bayes classifier ensemble</td>
<td>WCS</td>
</tr>
<tr>
<td>Carlson et al. [97], 2001</td>
<td>General</td>
<td>English</td>
<td>Golding and Roth [96]</td>
<td>WCS</td>
</tr>
<tr>
<td>Ruch et al. [98], 2001</td>
<td>General, medical</td>
<td>French</td>
<td>POS, WSD</td>
<td>Interactive</td>
</tr>
<tr>
<td>Golding and Roth [96], 1999</td>
<td>General</td>
<td>English</td>
<td>Winnow algorithm</td>
<td>WCS</td>
</tr>
<tr>
<td>Jones and Martin [99], 1997</td>
<td>General</td>
<td>English</td>
<td>LSA</td>
<td>WCS</td>
</tr>
<tr>
<td>Golding and Schabes [67], 1996</td>
<td>General</td>
<td>English</td>
<td>Naïve Bayes</td>
<td>WCS</td>
</tr>
</tbody>
</table>

Note: ANN, artificial neural network; HMM, hidden Markov model; LM, language model; LSA, latent semantic analysis; ME, maximum entropy; OCR, optical character recognition; POS, part-of-speech; SMT, statistical machine translation; WCS, word-confusion set; WSD, word-sense disambiguation.
The edit distance $D(s|w)$ of the incorrect word $s$ and a correction candidate $w$ in the a priori error model is a positive real number. In order to fit the probabilistic framework, it can be converted into the probabilistic framework by taking a negative logarithm \cite{100}:

$$P(s|w) = -\log D(s, w). \quad (3)$$

Methods of spelling correction in context are similar to morphological analysis, and it is possible to use similar methods of disambiguation from part-of-speech taggers in a context model of automatic spelling correction systems.

6.1. Language Model

The most common form of a language model is $n$-gram language model, calculated from the frequency of word sequences of size $n$. It gives the probability $P(w_i|w_{i-1}, \ldots, w_{i-(n-1)})$ of a candidate word given its history of $(n-1)$ words. If the given $n$-gram sequence is not presented in the training corpus, the probability is calculated by a back-off that considers shorter contexts. The $n$-gram language model only depends on previous words, but other classifiers can make use of arbitrary features in any part of the context. The language model is usually trained on a training corpus that represents language with correct spelling.

Neural language modeling brought new possibilities, as it can predict a word given arbitrary surrounding context. A neural network maps a word into a fixed-size embedding vector. Embedding vectors form a semantic space of words. Words that are close in the embedding space usually occur in the same context and are thus semantically close. This feature can be used in a spelling correction system to propose and rank a list of correction candidates \cite{63,101,102}.

6.2. Combination of Multiple Context Models

Context modeling often benefits from a combination of multiple statistical models. A spelling system proposed by Melero et al. \cite{73} used a linear combination of language models, each with a certain weight. Each language model can focus on a different feature: lowercase words, uppercase words, part-of-speech tags, and lemmas.

The authors in \cite{67} proposed a context model with multiple Bayesian classifiers. The first component of the context model is called “trigrams”. This system uses parts of speech as a feature for classification. The first part of the model assigns the highest probability to a candidate word and its context containing the most probable part-of-speech tags. The second part of the context model is a naïve Bayes classifier that takes the surrounding words and collocations (preceding word and current tag).

Another form of a statistical classifier for the context modeling with multiple models is the Winnow algorithm \cite{96,103}. This approach uses several Winnow classifiers trained with different parameters. The final rank is their weighted sum.

The model uses the same features (occurrence of a word in context and collocation of tags and surrounding word) as those in the previous approach \cite{67}. The paper by Golding and Roth \cite{96} was followed by Carlson et al. \cite{97}, which used a large-scale training corpus. Also, Li and Wang \cite{95} proposed a similar system for Chinese spelling correction.

An approach published by Banko and Brill \cite{90} proposed a voting scheme that utilized four classifiers. This approach focused on learning by using a large amount of data—over 100 million words. It uses a Winnow classifier, naïve Bayes classifier, perceptron, and a simple memory-based learner. Each classifier has a complementarity score defined by Brill et al. \cite{104} and is separately trained. The complementarity score indicates how accurate the classifier is.
6.3. Weighted Finite-State Transducers

If components of an ASC system (dictionary, error model, or context model) can be converted into a state machine, it is possible to create a single state machine by composing individual components. The idea of finite-state spelling was formalized by Pirinen and Lindén [7]. They compared finite-state automatic spelling correction systems with other conventional systems (Aspell and Hunspell) for English, Finnish, and Icelandic on the corpus of Wikipedia edits. They showed that this approach had comparable performance to that of others.

A weighted state transducer (WFST) is a generalization of a finite-state automaton, where each transcription rule has an input string, output string, and weight. One rule of the WFST system represents a single piece of knowledge about spelling correction—an edit operation of the error model or a probability of succeeding words in the context model.

Multiple WFSTs (dictionary, error model, and context model) can be composed into a single WFST by joining their state spaces and by removing useless states and transcription rules. After these three components are composed, the resulting transducer can be searched for the best path, which is the sequence of best-matching letters.

For example, the approach by Perez-Cortes et al. [105] took a set of hypotheses from the OCR. The output from OCR is an identity transducer (an automaton that transcribes the set of strings to the same set of strings) with weights on each transition that represents the probability of a character in the hypothesis. The character-level \( n \)-gram model represents a list of valid strings from the lexicon. The third component of the error model is a letter-confusion matrix calculated from the training corpus. The authors in [106,107] used handcrafted Arabic morphological rules to construct a WFST for automatic spelling correction.

A significant portion of text errors involves running together two or more words (e.g., ofthe) or splitting a single word (sp ent, th ebook) [2]. Weighted finite-state transducer (WFST) systems can identify word boundaries if the spacing is incorrect (http://openfst.org/twiki/bin/view/FST/FstExamples). However, inserting or deleting a space is still considered problematic because spaces have the annoying characteristic of not being handled by edit-distance operations [106].

7. Spelling Correction with Learning Error Model

The previous sections presented spelling correction systems with a fixed set of rules, prepared in advance by an expert. This section introduces approaches where the error model learns from a training corpus. The optimization algorithm iteratively updates the parameters of the error model (e.g., weights of the edit operations) to improve the quality of the ASC system.

A diagram in Figure 6 displays a structure of a learning error model. The algorithm for learning the error model uses the expectation-maximization procedure. A complete automatic spelling correction system contains a context model that is usually learned separately. The authors in [108] proposed to utilize the context model in the learning of the error model. Context probability is taken into account during the expectation step. Some approaches do not consider context at all. A comparison of approaches with the learning error model is shown in Tables 5 and 6.

![Figure 6. Spelling correction with learning error model](image-url)
Table 5. Spelling correction systems with learning of context model and error model—part I.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Application</th>
<th>Language</th>
<th>Context Model</th>
<th>Error Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Han et al. [109], 2019</td>
<td>General</td>
<td>Chinese</td>
<td>N/a</td>
<td>BiLSTM seq2seq [110]</td>
</tr>
<tr>
<td>Jain et al. [111], 2019</td>
<td>General</td>
<td>Hindi</td>
<td>LM</td>
<td>LD, LCM</td>
</tr>
<tr>
<td>Kinaci [24], 2019</td>
<td>General</td>
<td>Turkish</td>
<td>N/a</td>
<td>LSTM character LM</td>
</tr>
<tr>
<td>Lu et al. [112], 2019</td>
<td>Diacritic</td>
<td>Mongolian</td>
<td>N/a</td>
<td>Evolved transformer seq2seq</td>
</tr>
<tr>
<td>Mammadov [113], 2019</td>
<td>General</td>
<td>Azerbaijani</td>
<td>N/a</td>
<td>Seq2seq [110]</td>
</tr>
<tr>
<td>Roy [114], 2019</td>
<td>General</td>
<td>English</td>
<td>N/a</td>
<td>Seq2seq transformer</td>
</tr>
<tr>
<td>Yang et al. [115], 2019</td>
<td>Speech postprocessing</td>
<td>Chinese</td>
<td>N/a</td>
<td>CRF, seq2seq [110], BERT, character embeddings</td>
</tr>
<tr>
<td>Zaky and Romadhony [102], 2019</td>
<td>General</td>
<td>Indonesian</td>
<td>N/a</td>
<td>POS, word embeddings, BiLSTM seq2seq [110]</td>
</tr>
<tr>
<td>Zheng et al. [116], 2019</td>
<td>Speech postprocessing</td>
<td>Chinese</td>
<td>N/a</td>
<td>Transformer seq2seq</td>
</tr>
<tr>
<td>Zhou et al. [117], 2019</td>
<td>General</td>
<td>English</td>
<td>N/a</td>
<td>BiLSTM seq2seq [110]</td>
</tr>
<tr>
<td>Barteld et al. [118], 2018</td>
<td>Historical POS</td>
<td>Middle High German</td>
<td>N/a</td>
<td>Character LM</td>
</tr>
<tr>
<td>Sooraj et al. [119], 2018</td>
<td>General</td>
<td>Malayan</td>
<td>N/a</td>
<td>LSTM character LM</td>
</tr>
<tr>
<td>Sbattella and Tedesco [120], 2018</td>
<td>General</td>
<td>Italian</td>
<td>N/a</td>
<td>seq2seq LSTM</td>
</tr>
<tr>
<td>Fivez et al. [101], 2017</td>
<td>Medical</td>
<td>English, Dutch</td>
<td>Word embedding</td>
<td>DLD, double-Metaphone, character embeddings</td>
</tr>
<tr>
<td>Hladek et al. [8], 2016</td>
<td>OCR</td>
<td>English</td>
<td>HMM, LM</td>
<td>Ristad and Yianilos [100]</td>
</tr>
<tr>
<td>Silfverberg et al. [121], 2016</td>
<td>OCR</td>
<td>Finnish</td>
<td>N/a</td>
<td>WFST, Eger et al. [122], Lindén [123]</td>
</tr>
<tr>
<td>Abandah et al. [124], 2015</td>
<td>Diacritization</td>
<td>Arabic</td>
<td>N/a</td>
<td>Recurrent ANN</td>
</tr>
<tr>
<td>Hasan and Heger [125], 2015</td>
<td>Search query</td>
<td>English</td>
<td>LM</td>
<td>DLD, SMT</td>
</tr>
<tr>
<td>Lai et al. [126], 2015</td>
<td>General, medical</td>
<td>English</td>
<td>CRF for NER</td>
<td>Kernighan et al. [22]</td>
</tr>
<tr>
<td>Ramasamy et al. [127], 2015</td>
<td>General</td>
<td>Czech</td>
<td>Goldberg and Roth [96]</td>
<td>Church and Gale [128]</td>
</tr>
<tr>
<td>Evershed and Fitch [129], 2014</td>
<td>OCR</td>
<td>English</td>
<td>LM</td>
<td>LCM</td>
</tr>
<tr>
<td>Makazhanov et al. [130], 2014</td>
<td>General</td>
<td>Kazakh</td>
<td>N/a</td>
<td>Church and Gale [128]</td>
</tr>
</tbody>
</table>

Note: ANN, artificial neural network; BERT, bidirectional encoder representations from transformers; BiLSTM, bidirectional long short-term memory; CRF, conditional random fields; HMM, hidden Markov model; LCM, letter-confusion matrix; LSTM, long short-term memory; NER, named entity recognition; OCR, optical character recognition; POS, part-of-speech; seq2seq, sequence-to-sequence; WFST, weighted finite-state transducer.
Table 6. Spelling correction systems with learning of context model and error model—part II.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Application</th>
<th>Language</th>
<th>Context Model</th>
<th>Error Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mitankin et al. [131], 2014</td>
<td>OCR, historical</td>
<td>Old English</td>
<td>ME, LM</td>
<td>SMT</td>
</tr>
<tr>
<td>Sariev et al. [132], 2014</td>
<td>Historical text, OCR</td>
<td>Early Modern English, Bulgarian</td>
<td>SMT, LM, ME</td>
<td>LD, SMT</td>
</tr>
<tr>
<td>Wang et al. [133], 2014</td>
<td>Search query</td>
<td>English</td>
<td>N/a</td>
<td>ME</td>
</tr>
<tr>
<td>Huang et al. [134], 2013</td>
<td>General, automotive</td>
<td>English</td>
<td>N/a</td>
<td>Maximum of common characters, LD, ANN</td>
</tr>
<tr>
<td>Reffle and Ringlstetter [135], 2013</td>
<td>OCR, historical</td>
<td>Old German</td>
<td>LM</td>
<td>Bayes</td>
</tr>
<tr>
<td>Duan et al. [136], 2012</td>
<td>Search query</td>
<td>English</td>
<td>LM</td>
<td>SVM</td>
</tr>
<tr>
<td>Rashwan et al. [137], 2011</td>
<td>Diacritization</td>
<td>Arabic</td>
<td>LM</td>
<td>FSA, A star, ME</td>
</tr>
<tr>
<td>Perez-Cortes et al. [105], 2010</td>
<td>OCR, record linkage</td>
<td>Spanish</td>
<td>N/a</td>
<td>WFST, generalized LD, letter n-grams</td>
</tr>
<tr>
<td>Takasu [128], 2009</td>
<td>OCR</td>
<td>Japanese</td>
<td>N/a</td>
<td>Ristad and Yianilos [100], Takasu and Aihara [139]</td>
</tr>
<tr>
<td>Magdy and Darwish [140], 2008</td>
<td>OCR</td>
<td>Arabic</td>
<td>LM</td>
<td>LCM</td>
</tr>
<tr>
<td>Beaufort and Mancas-Thillou [141], 2007</td>
<td>OCR</td>
<td>English</td>
<td>WFST</td>
<td>LCM</td>
</tr>
<tr>
<td>Byun et al. [142], 2007</td>
<td>General</td>
<td>Korean</td>
<td>N/a</td>
<td>Learning general edit operations</td>
</tr>
<tr>
<td>Magdy and Darwish [143], 2006</td>
<td>OCR</td>
<td>Arabic</td>
<td>LM</td>
<td>Brill and Moore [29]</td>
</tr>
<tr>
<td>Oncina and Sebban [144], 2006</td>
<td>OCR</td>
<td>None</td>
<td>N/a</td>
<td>Ristad and Yianilos [100]</td>
</tr>
<tr>
<td>Ahmad and Kondrak [108], 2005</td>
<td>Search query</td>
<td>English</td>
<td>LM</td>
<td>Ristad and Yianilos [100]</td>
</tr>
<tr>
<td>Toutanova and Moore [6], 2002</td>
<td>General</td>
<td>English</td>
<td>N/a</td>
<td>Brill and Moore [29]</td>
</tr>
<tr>
<td>Brill and Moore [29], 2000</td>
<td>General</td>
<td>English</td>
<td>LM</td>
<td>DLD, extended Church and Gale [128]</td>
</tr>
<tr>
<td>Ristad and Yianilos [100], 1998</td>
<td>General</td>
<td>English</td>
<td>N/a</td>
<td>LCM</td>
</tr>
<tr>
<td>Church and Gale [128], 1991</td>
<td>General</td>
<td>English</td>
<td>N/a</td>
<td>Four LCMs</td>
</tr>
<tr>
<td>Kernighan et al. [22], 1990</td>
<td>General</td>
<td>English</td>
<td>N/a</td>
<td>Four LCMs</td>
</tr>
</tbody>
</table>

Note: ANN, artificial neural network; FSA, finite-state automaton; LCM, letter-confusion matrix; LM, language model; LSTM, long short-term memory; ME, maximum entropy; OCR, optical character recognition; seq2seq, sequence-to-sequence; SMT, statistical machine translation; SVM, support vector machine; WFST, weighted finite state transducer.
ASC systems with a learning error model often complement optical character recognition systems (OCR). The digitized document contains spelling errors characteristic of the quality of the paper, scanner, and OCR algorithm. If the training database (original and corrected documents) is large enough, the spelling system is adapted to the data. A training sample from the TREC-5 confusion track corpus [9] is displayed in Figure 7.

Correct: bulletin
Incorrect: bM.etin, bWetin, bMetinh, bUletin
Count: 2 2 4 23

Figure 7. Example misspellings of word the “bulletin” from optical character recognition (OCR).

7.1. Word-Confusion Set

The simplest method of estimating the learning error model is a word-confusion set that counts the cooccurrences of correct and incorrect words in the training corpus. It considers a pair of correct and incorrect words as one big edit operation. The word-confusion set remembers possible corrections for each frequently misspelled form (See Figure 7). This method was used by Gong et al. [145] to improve the precision of e-mail spam detection.

Its advantages are that it can be easily created and manually checked. The disadvantage of this simple approach is that it is not possible to obtain a corpus that has every possible misspelling for every possible word. The second problem of the word-confusion set is that error probabilities are far from “real” probabilities because training data are always sparse. Shannon’s theorem states that it is not possible to be 100% accurate in spelling correction.

7.2. Learning String Metrics

The sparseness problems of the word-confusion set are solved by observing smaller subword units (such as letters or morphemes). For example, Makazhanov et al. [130] utilized information about morphemes in the Kazakh language to improve automatic spelling correction. The smallest possible subword units are letters. Estimating parameters of edit operations partially mitigates the sparseness problem because smaller sequences appear in the training corpus more frequently. The authors in [29] presented an error model that learned general edit operations. The antecedent and consequent parts of the edit operations can be arbitrary strings called partitions. The partition of the strings defines the edit operations.

Generalized edit distance is another form of a learning error model. The antecedent and consequent part of an edit operation is a single symbol that can be a letter or a special deletion mark. Edit distance is generalized by considering the arbitrary weight of an operation. Weights of each possible edit operation of the Levenshtein distance (LD) can be stored in a single letter-confusion matrix (LCM). \( \Delta \) weights for generalized edit distance are stored in four matrices [128]. The generalized edit distance is not always a metric in the strict mathematical sense because the distance in the opposite direction can be different. More theory about learning string metrics can be found in a book [146] or in a survey ([147], Section 5.1).

Weights \( \Delta \) in an LCM express the weight of error types (Figure 8). If the LCM is a matrix of ones with zeros on the main diagonal, it expresses the Levenshtein edit distance. Each edit operation has a value of 1, and the sum of edit operations is the Levenshtein edit distance. The edit distance with weights is calculated by a dynamic algorithm [53,148].

The LCM for a Levenshtein-like edit distance can be estimated with an expectation-maximization algorithm [100]. The learning algorithm calculates weights of operations for each training sample that are summed and normalized to form an updated letter confusion matrix.

If the training corpus is sparse (which it almost always is), the learning process brings the problem of overfitting. Hládek et al. [8] proposed a method for smoothing parameters in a letter-confusion matrix. Bilenko and Mooney [149] extended string-distance learning with an affine gap penalty.
(allowing for random sequences of characters to be skipped). Also, Kim and Park [150] presented an algorithm for learning a letter-confusion matrix and for calculating generalized edit distance. This algorithm was further extended by Hyyrö et al. [151].

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>d</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.2</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>b</td>
<td>0.2</td>
<td>0.5</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>c</td>
<td>0.2</td>
<td>0.1</td>
<td>0.6</td>
<td>0.3</td>
</tr>
<tr>
<td>d</td>
<td>0.2</td>
<td>0.1</td>
<td>0.2</td>
<td>0.8</td>
</tr>
</tbody>
</table>

**Figure 8.** Example of a letter-confusion matrix for the alphabet of symbols a, b, c, and d for Levenshtein distance (left) and arbitrary letter confusion matrix (right): the matrix gives a weight of transcription of the letter in the vertical axis to the letter in the horizontal axis.

7.3. Spelling Correction as Machine Translation of Letters

Spelling correction can be formulated as a problem of searching for the best transcription of an arbitrary sequence of symbols into another sequence. This type of problem can be solved with methods typical for machine translation. General string-to-string translation models are not restricted to the spelling error correction task but can also be applied to many problems, such as grapheme-to-phoneme conversion, transliteration, or lemmatization [122]. The machine-translation representation of the ASC overcomes the problem of joined and split words but requires a large corpus to properly learn the error model.

Zhou et al. [117] defined the machine-translation approach to spelling correction by the following equation:

\[
s' = \arg \max_s P(s|S),
\]

where \(S\) is the given incorrect sequence, \(s\) is the possibly correct sequence, and \(s'\) is the best correction.

Characters are “words” of “correct” and “incorrect” language. Words in the training database are converted into sequences of lowercase characters, and white spaces are converted into special characters. The machine-translation system is trained on a parallel corpus of examples of spelling errors and corrections. Sariev et al. [132] and Koehn et al. [152] proposed an ASC system that utilizes a statistical machine-translation system called Moses (http://www.statmt.org/moses/).

The authors in [125] cast spelling correction into the machine translation of character bigrams. The spelling system is trained on logs of search queries. It was assumed that the corrections of queries by the user follow misspelled queries. This heuristics creates a training database. To improve precision, character bigrams are used instead of single characters.

Statistical machine-translation models based on string alignment, translation phrases, and \(n\)-gram language models are replaced by neural machine-translation systems. The basic neural-translation architecture, based on a neural encoder and decoder, was proposed by Sutskever et al. [110]. The translation model learns \(P(y_1...y_T|x_1...x_T)\) by encoding the given sequence into a fixed-size vector [117]:

\[
s = f_e(x_1, ..., x_T) = h_T.
\]

The sequence-embedding vector is decoded into another sequence by a neural decoder [117]:

\[
y_t = f_d(s, y_1, ..., y_{t-1}) = h_T.
\]

The decoder takes the encoded vector language model and generates the output. Zhou et al. [117] showed that, by using \(k\)-best decoding in the string-to-string translation models, they achieved much better results on the spelling correction task than those of the three baselines, namely edit distance, weighted edit distance, and the Brill and Moore model [104].
8. Evaluation Methods

The development of automatic spelling correction systems requires a way to objectively assess the results. It is clear though that it is impossible to propose a “general” spelling benchmark because the problem is language- and application-dependent.

Three possible groups of methods exist for evaluating automatic spelling correction:

- accuracy, precision, and recall (classification);
- bilingual-evaluation-understudy (BLEU) score (machine translation); and
- mean reciprocal rank and mean average precision (information retrieval).

The most common evaluation metrics is classification accuracy. The disadvantage of this method is that only the best candidate from the suggestion list is considered, and order and count of the other proposed correction candidates are insignificant. Therefore, it is not suitable for evaluating an interactive system.

Automatic spelling correction is similar to machine translation. A source text containing errors is translated to its most probable correct form. The approach takes the whole resulting sentence, and it is also convenient for evaluating the correction of a poor writing style and non-word errors. It was used by Sariev et al. [132], Gerdjikov et al. [153] and Mitankin et al. [131].

Machine-translation systems are evaluated using the BLEU score, which was first proposed by Papineni et al. [154]:

“The task of a BLEU implementation is to compare $n$-grams of the candidate with the $n$-grams of the reference translation and to count the number of matches. These matches are position-independent. The more matches, the better the candidate translation.”

The process of automatic spelling correction is also similar to information retrieval. An incorrect word is a query, and the sorted list of the correction candidates is the response. This approach evaluates the whole list of suggestions and favors small lists of good (highly ranked) candidates for correction. The two following evaluation methodologies are used to evaluate spelling:

- Mean reciprocal rank: A statistical measure for evaluating any process that produces a list of possible responses to a sample of queries ordered by the probability of correctness. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer. The mean reciprocal rank is the average of the reciprocal ranks of results for a sample of queries [155].
- Mean average precision: Average precision observes how many times a correct suggestion is on the $n$-th place or better in a candidate list [40]. It is calculated from average precision for $n$ in the range from 1 to $k$ ($k$ is a constant, e.g., 10).

Machine translation and information retrieval are well-suited for evaluating interactive systems because they consider the whole candidate list. A smaller candidate list is more natural to comprehend. The best correction can be selected faster from fewer words. On the other hand, the candidate list must be large enough to contain the correct answer.

8.1. Evaluation Corpora and Benchmarks

Several authors proposed corpora for specific tasks and languages, but no approach was broadly accepted. The authors in [12] proposed the Koran as a benchmark for the evaluation of Arabic diacritizations. Reynaert [156] presented an XML format and OCR-processed historical document set in Dutch for the evaluation of automatic spelling correction systems.

The most used evaluation set for automatic spelling correction of OCR is TREC-5 Confusion Track [9]. It was created by scanning a set of paper documents. The database consists of original and recognized documents, so it is possible to identify correct–incorrect pairs for system training and

Also, Hagen et al. [34] proposed a corpus of corrected search queries in English (https://www.uni-weimar.de/en/media/chairs/computer-science-and-media/webis/corpora), and provided an evaluation metric. They re-implemented the best-performing approach [157] from the Microsoft Speller Challenge (https://github.com/webis-de/SIGIR-17).

Tseng et al. [158] presented a complete publicly available spelling benchmark for the Chinese language, preceded by Wu et al. [159]. Similarly, the first competition on automatic spelling correction for Russian was published by Sorokin et al. [160].

8.2. Performance Comparison

Table 7 gives a general overview of the performance of automatic spelling correction systems. It lists approaches with well-defined evaluation experiments performed by the authors. The table displays the best value reached in the evaluation and summarizes the evaluation corpora. Only a few corpora were available that are suitable for evaluating an ASC system (such as TREC-5).

It is virtually impossible to compare the performance of state-of-the-art spelling correction systems. Each author solves a different task and uses their methodology, custom testing set, and various evaluation corpora with different languages. The displayed values cannot be used for mutual comparison but are instead a guide for selecting an evaluation method. A solution would be a spelling correction toolkit that implements state-of-the-art methods for error modeling and context classification. A standard set of tools would allow for comparison of individual components, such as error models.
Table 7. Reported evaluation results.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Evaluation (%)</th>
<th>Test Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azmi et al. [60], 2019</td>
<td>A 98, F 90.7, P 83.5, R 99.2</td>
<td>Arabic Newspaper Corpora</td>
</tr>
<tr>
<td>Han et al. [109], 2019</td>
<td>A 62.67, F 49.33, P 81.12, R 36.33</td>
<td>Tseng et al. [158], Wu et al. [159]</td>
</tr>
<tr>
<td>Lv et al. [72], 2016</td>
<td>A 95.72, F 95.78</td>
<td>Chinese OCR Medical Records</td>
</tr>
<tr>
<td>Melero et al. [73], 2016</td>
<td>P 82.56</td>
<td>Twitter texts (baseline 56.88%)</td>
</tr>
<tr>
<td>Attia et al. [107], 2015</td>
<td>A 93.64</td>
<td>Arabic Gigaword Corpus 5th Edition</td>
</tr>
<tr>
<td>Lai et al. [126], 2015</td>
<td>A 88.2, F 94.4, P 96.2, R 92.7</td>
<td>Clinical Notes of Patients</td>
</tr>
<tr>
<td>Ramasamy et al. [127], 2015</td>
<td>F 95.4, P 95.0, R 95.9</td>
<td>WebColl, CzeSL-MAN, Czech National Corpus: SYN2005 and SYN2010</td>
</tr>
<tr>
<td>Evershed and Fitch [129], 2014</td>
<td>W 6.4</td>
<td>Sydney Morning Herald, 1842-1954</td>
</tr>
<tr>
<td>Mitankin et al. [131], 2014</td>
<td>A 93.96</td>
<td>1641 Depositions Old English</td>
</tr>
<tr>
<td>Sariev et al. [132], 2014</td>
<td>W 16.84/4.98/4.25/3.27</td>
<td>ICAMET/IMPACT BG/1641 Deposition/TREC-5</td>
</tr>
<tr>
<td>Sagiadinos et al. [79], 2014</td>
<td>F 97.4, P 97.8, R 97.0</td>
<td>Eleftherotypia—The Modern Greek Text Corpus</td>
</tr>
<tr>
<td>Wang et al. [133], 2014</td>
<td>F 85.89</td>
<td>Microsoft Speller Challenge</td>
</tr>
<tr>
<td>Ehsan and Faili [80], 2013</td>
<td>F 36, P 56, R 31</td>
<td>Persian Corpus Peykareh</td>
</tr>
<tr>
<td>Duan et al. [136], 2012</td>
<td>F 94.9/92.8, P 96.3/90.3, R 94.4/95.3</td>
<td>TREC-5/MS Microsoft Speller Challenge</td>
</tr>
<tr>
<td>Flor [59], 2012</td>
<td>F 85.87, P 85.50, R 86.25</td>
<td>ETS Spelling Corpus</td>
</tr>
<tr>
<td>Sha et al. [42], 2011</td>
<td>A 93.3</td>
<td>User Behavior Records in Real Online Study Website Chinese</td>
</tr>
<tr>
<td>Stüker et al. [85], 2011</td>
<td>W 9.7</td>
<td>The Fay Database—Children’s Free Writing German</td>
</tr>
<tr>
<td>Wong and Glance [86], 2011</td>
<td>A 88.73</td>
<td>Clinical Progress Notes (<a href="http://physionet.org">http://physionet.org</a>)</td>
</tr>
<tr>
<td>Takasu [138], 2009</td>
<td>A 94.2</td>
<td>1000 Japanese Articles</td>
</tr>
<tr>
<td>Beaufort and Mancas-Thillou [141], 2007</td>
<td>A 65.4</td>
<td>English ICDAR 2003 Corpus</td>
</tr>
<tr>
<td>Byun et al. [142], 2007</td>
<td>A 92.75</td>
<td>SMS messages in Korean</td>
</tr>
<tr>
<td>Carlson and Fette [89], 2007</td>
<td>A 95.2/95.8</td>
<td>Brown Corpus/Wall Street Journal Corpus</td>
</tr>
<tr>
<td>van Delden et al. [46], 2004</td>
<td>A 93.3</td>
<td>Misspellings from two NASA databases, Structural and Fuel Cells</td>
</tr>
<tr>
<td>Héja and Surján [92], 2003</td>
<td>P 37.2, R 82.6</td>
<td>Corpus of 92 Clinical Diagnoses in Hungarian</td>
</tr>
</tbody>
</table>

Note: A, accuracy; F, F-measure/F1-score; P, precision; R, recall; W, word error rate.
9. Conclusions

The chronological sorting and grouping of the summary tables with references in this work reveal several findings. The research since the last comprehensive survey [2] brought new methods for spelling correction. On the other hand, we can say that the progress of spelling correction in all areas was slow until the introduction of deep neural networks.

New, a priori spelling correction systems are often presented for low-resource languages. Authors propose rules for a priori error model that extend the existing phonetic algorithm or adjust the edit distance for the specifics of the given language.

Spelling correction systems in context are mostly proposed for languages with sufficient language resources for language modeling. Most of them use n-gram language models, but some approaches use neural networks or other classifiers. Scientific contributions for spelling in context explore various context features with statistical classifiers.

Spelling correction with the learning error model shows the biggest progress. The attention of the researchers moves from statistical estimation of the letter confusion matrices to utilization of the statistical machine translation.

This trend is visible mainly in Tables 5 and 6, where we can observe the growing popularity of the use of encoder–decoder architecture and deep neural networks since 2018. New approaches move from word-level correction to arbitrary character sequence correction because new methods based on deep neural networks bring better possibilities. Methods based on machine translation and deep learning solve the weakest points of the ASC systems, such as language-specific rules, real-word errors, and spelling errors with spaces. The neural networks can be trained on already available large textual corpora.

The definition of the spelling correction stated in the Equation (1) begins to be outdated because of the new methods. Classical statistical models of context-based (n-gram, log-linear regression, and naïve Bayes classifier) on the presence of word-level features in the context are no longer important. Instead, feature extraction is left to the hidden layers of the deep neural network. The correction of spelling errors becomes the task of transcribing a sequence of characters to another sequence of characters using a neural network, as it is stated in Equation (4). Research in the field of spelling error correction thus approaches other solutions to other tasks of speech and language processing, such as machine translation or fluent speech recognition.

On the other hand, the scientific progress of learning error models is restricted by the lack of training corpora and evaluation benchmarks. Our examination of the literature shows that there is no consensus on how to evaluate and compare spelling correction systems. Instead, almost every paper uses its own evaluation set and evaluation methodology. In our opinion, the reason is that most of the spelling approaches strongly depend on the specifics of the language and are hard to adapt to another language or a different application. Recent algorithms based on deep neural networks are not language dependent, but their weak point is that they require a large training set, often with expensive manual annotation. These open issues call for new research in automatic spelling correction.

Author Contributions: Conceptualization, D.H.; methodology, D.H.; formal analysis, J.S.; investigation, D.H.; resources, D.H.; writing—original draft preparation, D.H.; writing—review and editing, M.P. and J.S.; supervision, M.P.; project administration, J.S.; funding acquisition, M.P. All authors have read and agreed to the published version of the manuscript.

Funding: Research in this paper was supported by the Slovak Research and Development Agency (Agentúra na podporu výskumu a vývoja) under projects APVV-15-0517 and APVV-15-0731; the Scientific Grant Agency (Vedecká grantová agentúra MŠVVaS SR a SAV), project number VEGA 1/0753/20; and the Cultural and Educational Grant Agency (Kultúrna a edukačná grantová agentúra MŠVVaS SR), project number KEGA 009TUGE-4-2019, both funded by the Ministry of Education, Science, Research, and Sport of the Slovak Republic.

Acknowledgments: The authors want to thank Jozef Juhár for the team leadership, and personal and financial support.

Conflicts of Interest: The authors declare no conflict of interest.
Abbreviations

The following abbreviations were used in this manuscript:

ANN Artificial neural network
ASC Automatic spelling correction
CFG Context-free grammar
CRF Conditional random fields
DLD Damerau–Levenshtein distance
EM Expectation maximization
FSA Finite-state automaton
HMM Hidden Markov model
IR Information retrieval
k-NN k-nearest neighbors
LCS Longest common subsequence
LD Levenshtein distance
LCM Learning letter-confusion matrix
LM Language model
LR Linear regression
LSA Latent semantic analysis
LSTM Long short-term memory
seq2seq Sequence-to-sequence
ME Maximum entropy
OCR Optical character recognition
PMI Pointwise mutual information
POS Part-of-speech tagging
RF Random forests
SMT Statistical machine translation
SVM Support vector machine
WFST Weighted finite-state transducer
WSD Word-sense disambiguation
WCS Word-confusion set

References


25. Brill, E.; Moore, R.C. An improved error model for noisy channel spelling correction. In Proceedings of the 38th Annual Meeting on Association for Computational Linguistics ACL 00, Hong Kong, China, 7 October 2000; pp. 286–293. [CrossRef]


34. Hagen, M.; Potthast, M.; Gohsen, M.; Rathgeber, A.; Stein, B. A large-scale query spelling correction corpus. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017. [CrossRef]


37. Rees, T. Taxamatch, an algorithm for near (‘Fuzzy’) matching of scientific names in taxonomic databases. PLoS ONE 2014, 9, e107510. [CrossRef]


115. Yang, L.; Li, Y.; Wang, J.; Tang, Z. Post text processing of chinese speech recognition based on bidirectional LSTM networks and CRF. *Electronics* 2019, 8, 1249. [CrossRef]


123. Lindén, K. Multilingual modeling of cross-lingual spelling variants. *Inf. Retr.* 2006. [CrossRef]


© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).