SpellInk: Interactive correction of spelling mistakes in handwritten text

Konstantin KUZNETSOV, Michael BARZ, Daniel SONNTAG
German Research Center for Artificial Intelligence (DFKI)
Saarbrücken, Germany

Abstract. Despite the current dominance of typed text, writing by hand remains the most natural mean of written communication and information keeping. Still, digital pen input provides limited user experience and lacks flexibility, as most of the manipulations are performed on a digitalized version of the text. In this paper, we present our prototype that enables spellchecking for handwritten text: it allows users to interactively correct misspellings directly in a handwritten script. We plan to study the usability of the proposed user interface and its acceptance by users. Also, we aim to investigate how user feedback can be used to incrementally improve the underlying recognition models.

Keywords. digital pen, handwriting recognition, handwriting generation

Introduction

Handwriting continues to be one of the most popular means of communication. It has been shown to be more efficient than keyboard input [1], useful in cognitive assessment [2], as well as beneficial in various learning tasks [3,4,5]. It is not uncommon for written text to contain spelling errors which users would like to fix, better without rewriting their own samples. Although spelling correction has been studied for decades [6], only a few papers addressed this problem for handwritten scripts [7,8]. In that case, the user could continuously use the writing device without the need to switch between handwritten and digital representation to correct errors. This task requires a combination of handwriting recognition [9,10], spell checking [11], and handwriting synthesis [12,7] techniques. In this demo, we present SpellInk[1]: a mobile application which allows users to take handwritten notes, immediately spellcheck and interactively correct them, while the user's writing style is preserved. The system runs on any Android tablet or smartphone and performs all processing on the device, thus preserving the user's privacy.

Spelling Correction Application

The main screen of our prototype contains a canvas where the user can enter notes with a digital pen. After a user writes a word by hand, the system checks

[1] `https://github.com/DFKI-Interactive-Machine-Learning/spellink`

it for spelling mistakes. If one is identified, the erroneous word is underlined with a double red line. Tapping on it with a pen opens a context menu showing to the user a list of corrections (see Figure 1). Upon selection, the correct word is transformed into a handwritten form, preserving the user's style, to substitute the incorrect word.

Our text recognition module relies on the code by Daniel Vorberg[2], which we reimplemented in Kotlin and TensorFlow Mobile for being used on Android mobile devices. It follows the approach to online handwriting recognition proposed by Liwicki et al. [13], augmented with a specifically designed set of features. The underlying model is based on a bidirectional LSTM neural network with a Connectionist Temporal Classification (CTC) loss function and a CTC beam search decoder. To process the pen input, we first normalize pen coordinates of strokes (the slant, dimensions, and the baseline are adjusted) and transform them into a sequence of cubic B-splines. Next, using this representation and temporal data, we compute a set of features, which are then fed to the pre-trained model. While the model training has been performed on a server using the IAM-OnDB dataset [14], the inference runs on the mobile device such that no sensitive user data is sent outside.

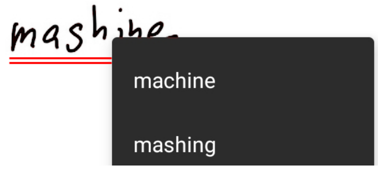

Figure 1.: By tapping a word, marked as erroneous, the user invokes a menu with suggested corrections.

When the results of the handwriting recognition become available, they are passed to the spellchecking service. We built it on top of the well-known *Hunspell* package [15], which we compiled into a native library using Android NDK. If a written word is not in the dictionary of known words, our service generates a list of corrections, which will be shown to the user upon request (see Figure 1). To seamlessly integrate the selected correction in the text, it should be converted to a handwriting script that reflects the user's style. We use the recurrent neural network proposed by Graves et al. [12] to generate a handwritten version of the correct word for replacing the incorrect one. As a backbone we use the *handwriting-synthesis* project[3], which we reimplemented in TensorFlow Mobile. As outlined by the authors, this model often fails in reliably generating handwritings for styles that are not presented in the training data. Since retraining is very expensive, we personalize the generation by selecting the most similar style from a set of 200 available styles. To compare them with the user's style, we use the Mahalanobis distance calculated on digital pen features by Rubine [16]. This way, SpellInk substitutes the incorrect word with a generated instance of the selected correct word that preserves the user's writing style.

Next, we plan to study the usability of the proposed user interface and its acceptance by users. Besides, we propose to investigate how user feedback can be used to incrementally improve the underlying recognition and generation models.

[2] `https://github.com/antemons/smart-manuscript`
[3] `https://github.com/sjvasquez/handwriting-synthesis`

References

[1] Mueller PA, Oppenheimer DM. The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. Psychological science. 2014;25(6):1159-68.

[2] Prange A, Sonntag D. Modeling Users' Cognitive Performance Using Digital Pen Features. Frontiers in Artificial Intelligence. 2022;5.

[3] Anthony L, Yang J, Koedinger KR. A paradigm for handwriting-based intelligent tutors. International Journal of Human-Computer Studies. 2012;70(11):866-87.

[4] Bezine H. Authoring mobile tool for improving cursive handwriting learning/writing skills. International Journal of Learning Technology. 2020;15(2):150-79.

[5] Barz M, Altmeyer K, Malone S, Lauer L, Sonntag D. Digital pen features predict task difficulty and user performance of cognitive tests. In: Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization; 2020. p. 23-32.

[6] Hládek D, Staš J, Pleva M. Survey of automatic spelling correction. Electronics. 2020;9(10):1670.

[7] Aksan E, Pece F, Hilliges O. Deepwriting: Making digital ink editable via deep generative modeling. In: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems; 2018. p. 1-14.

[8] Maksai A, Rowley H, Berent J, Musat C. Inkorrect: Online Handwriting Spelling Correction. arXiv; 2022. Available from: https://arxiv.org/abs/2202.13794.

[9] Carbune V, Gonnet P, Deselaers T, Rowley HA, Daryin A, Calvo M, et al. Fast multi-language LSTM-based online handwriting recognition. International Journal on Document Analysis and Recognition (IJDAR). 2020;23(2):89-102.

[10] Frinken V, Uchida S. Deep BLSTM neural networks for unconstrained continuous handwritten text recognition. In: 2015 13th international conference on document analysis and recognition (ICDAR). IEEE; 2015. p. 911-5.

[11] Näther M. An In-Depth Comparison of 14 Spelling Correction Tools on a Common Benchmark. In: Proceedings of the 12th Language Resources and Evaluation Conference; 2020. p. 1849-57.

[12] Graves A. Generating sequences with recurrent neural networks. arXiv preprint arXiv:13080850. 2013.

[13] Liwicki M, Graves A, Fernàndez S, Bunke H, Schmidhuber J. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In: Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007; 2007. .

[14] Liwicki M, Bunke H. IAM-OnDB-an on-line English sentence database acquired from handwritten text on a whiteboard. In: Eighth International Conference on Document Analysis and Recognition (ICDAR'05). IEEE; 2005. p. 956-61.

[15] Németh L. Hunspell;. Available from: http://hunspell.github.io [cited 01.03.2022].

[16] Rubine D. Specifying gestures by example. ACM SIGGRAPH computer graphics. 1991;25(4):329-37.