# Multi-Stage Deep Learning for Context-Free Handwriting Recognition

Daniel Lückehe[1], Nicole Mühlpforte[2], Alfred O. Effenberg[2], and Gabriele von Voigt[1]

[1] Computational Health Informatics, Leibniz University Hanover, Germany
lueckehe@chi.uni-hannover.de,
[2] Institute for Sports Science, Leibniz University Hanover, Germany

**Abstract.** Handwriting recognition approaches usually use the semantic context of the individual letters. This helps to achieve highly accurate classification results. As it is less likely to detect semantically invalid letters, these approaches imply an error correction. In most cases, this is a beneficial side-effect but it makes the approaches not usable if these semantic invalid letters should be detected, e.g., in the case of educational software where spelling mistakes should be recognized.
For this purpose, we developed an advanced context-free handwriting recognition which is based on multiple techniques from the field of deep learning. We motivate the individual components of our approach and show how the components can benefit from each other. In the experimental section, the behavior of our new approach is analyzed in detail and the classification accuracy is compared between the different approaches.

**Keywords:** Handwriting Recognition, Deep Learning, Human-Machine Interface, Learning Support, Educational Software

## 1 Introduction

Writing is a key skill for learning results at school and professional success [8]. Thereby, correct orthography is an important part. In school, it is often trained by handwritten exercises as this can improve the orthography skills. Thus, handwriting is a crucial skill.

To realize educational software which supports the development of orthography, especially for children with spelling problems [22], a well-working handwriting recognition is required. Usually, the objective of a handwriting recognition is to be as accurate as possible. Therefore, the semantic context of the individual letters is used as this can improve the accuracy. For example, it might be challenging to distinguish between the written letters $u$ and $v$. But if the letters are considered in their semantic context, the task may be easy: in the middle of the word *house*, there is an $u$. So, if a handwriting recognition algorithm is not able to determine whether there is an $u$ or $v$ in the middle of the word, the algorithm can determine the letter by its context easily. Even a wrongly written *hovse* would be recognized as *house*. In practice, this increases the accuracy of

handwriting recognition algorithms significantly and the implicit error correction is mostly a beneficial side-effect. But in the case of educational software for children all errors must be detected to provide valuable assistance. Thus, in this case, a usual handwriting recognition algorithm is not useable.

There are multiple ways of taking spelling mistakes into account. For example, the software could display the error. As too much shown mistakes might be demotivating, the software could also collect all errors and give a report to a teacher who can individually support the child. Also, it is possible that the educational software adapts to the child. All spelling mistakes can be classified, e.g., there are errors with double consonants after a short vowel. If a child makes a disproportionately high number of errors of a certain class, this class could be more practiced. For all these ways taking spelling mistakes into account, a handwriting recognition is required which does not use the semantic context.

Our paper is structured as follows. In the following, the foundation of the paper is laid and the data set is introduced. Then, we present two different perspectives to the data: a visual one and a movement oriented one. In Section 3, we introduce our new approach of a context-free handwriting recognition which combines both perspectives. In the experimental results, shown in Section 4, the capabilities of our approach are presented and analyzed. Thereby, the capabilities of the single perspectives are compared to our approach. In the last section, conclusions are drawn.

### 1.1 Foundation

This work is based on a project in which the handwriting of elementary school children was recorded to develop a new educational software for children to train spelling skills. There are already software products to train spelling skills like `Gut1` [7] or `Tintenklex` [4] but to the best of our knowledge, none of them includes a handwriting recognition, i.e., they are all using a keyboard input.

To develop a handwriting recognition for an educational software, 189 children from the first to the fourth grade have done a handwriting task in four elementary schools in the region of Hanover, Germany. In this task, each child wrote the whole alphabet of small and capital letters and the German special characters. Additionally, each child wrote complete words which contains all possible letters. The words are selected from a list. There was no dictation and the processing sequence was freely chosen. The handwriting was recorded on an iPad Air using an Adonit Jot Pro stylus. Afterward, the individual letters have been separated and two times classified by two research assistants.

These data have been used in [19] to develop a handwriting recognition using a high-level representation. It is a nice approach but due to its poor classification accuracy, it is not useable. In the experimental section of their paper, only five capital letters are employed – the letters *A*, *M*, *O*, *T*, *U*. These letters are correctly predicted in 74.7% of the cases. The limitation of five letters and a classification error of more than 25% are not acceptable. The objective must be to use all letters and to achieve a classification accuracy of significantly above 90%. Otherwise, the errors of the recognition will demotivate the children while

using it. Especially, correctly written words which are classified as an error are frustrating. So, the recognition must work in the context of an educational software and thereby, the focus must be put to the task of the educational software and must not be put to errors of the handwriting recognition.

### 1.2 Data Set

In this section, we introduce the employed data set. It contains 34 627 samples which are not equally distributed. The frequency results from the occurrence while writing. Thus, there are more small than capital letters in the data set. The most frequent letter is the small $e$ which occurs 4205 times. It is followed by the small $n$, 2596 times, and the small $r$, 1518 times. The median occurrence of a letter is 336 times and even the least frequent one occurs 179 times. It is the capital special character $\ddot{U}$. Overall, there are 59 different letters: 26 capital and 26 small letters, capital and small special characters (the umlauts: ä, ö, ü), and another special character (eszett: ß). As shown in Figure 1, the letters in the data set are written in different writing styles. The figure presents six different ways to write a small $t$. Additionally, there is a large variation in the letters as they are written by children.

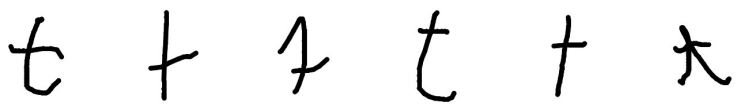


**Fig. 1.** Example of six written small letters $t$ visualizing different writing styles

The data set includes a chronological sequence for each written letter. For each time step, the $x$- and $y$-position of the pen which is used for the writing are given. Additionally, a flag is set if the pen is lifted. Based on this data, the letter should be classified. Thereby, it is interesting that the data can be interpreted as visual images. As the children grasp the written letters visually, this approach is obvious. Additionally, the data can be interpreted as a writing movement. We present both interpretations in the next section and show the advantages and disadvantages of them. Then, in our new approach, we employ both interpretations at the same time to combine their advantages.

## 2 Applying One Perspective to the Data

In this section, we show two interpretations of the data. In both cases, we employ methods from the field of deep learning to classify the written letters. In the first perspective, the data are interpreted as visual images. These images can be classified by a Convolutional Neural Network (CNN) [13, 14]. The second

perspective is taking the writing movement into account. As the movement is a time series, a Long Short-Term Memory Network (LSTM) [9, 11] is well suited to classify letters based on this perspective.

*Deep Visual Approach* In our deep visual approach, the data are interpreted as visual images. This is the most obvious perspective, as the children grasp their written letters visually. Based on the data, we create images with a resolution of $296 \times 220$ pixels. As fine structures are not relevant for the classification, the image sizes are reduced by the factor 4.

We choose a CNN for the classification task. CNNs have been very successful in the last years [18]. They are deep neural networks applying a structure of neurons which is inspired by the human visual cortex [1]. Therefore, they are well suited to classify images. The structure of our approach applying a CNN is very straight-forward. It is shown in Figure 2. The data are interpreted from the visual perspective resulting to $\mathbf{x}_{\text{visual}}$. One pattern $\mathbf{x}_{\text{visual}}$ contains a brightness information for each entry of a matrix with the size of $296 \times 220$ representing the pixels. The pattern is handed to the CNN which classifies it. The classification result is $y$.

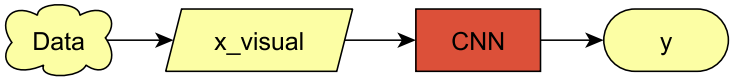



**Fig. 2.** Structure of our deep visual approach

The used CNN employs a classical structure [12]. The first layer is a convolutional layer employing 32 filters with the size of $3 \times 3$ pixels. It is followed by a max pooling layer which downsamples the data. We employ a $2 \times 2$ downsampling, i.e., the data height and width are reduced by the factor 2. Then, the combination of a convolutional layer and a max pooling layer is repeated but now, the convolutional layer employs 64 filters. The extracted features from the convolutional and a max pooling layers are handed to a dense layer. As activation function, we employ the ReLU function [16] for all the described layers. To avoid overfitting, a dropout layer [21] is used. Finally, an output layer with 59 neurons determines the classification results for the 59 different letters. The output layer uses the softmax function [2].

*Deep Movement Approach* Our second approach is to interpret the data as a movement. This means movement vectors are determined based on the movement of the pen. Multiple vectors yield a complete letter. Each child has its own writing speed which is not relevant for the classification. Thus, all letters are normalized to a size of 32 movement vectors. As the vectors can have different length, the relative speed while writing a letter is taken into account. Besides $\Delta x$ and $\Delta y$ values which describe the movement, there is a binary information for each vector. This information indicates whether the pen is lifted or not.

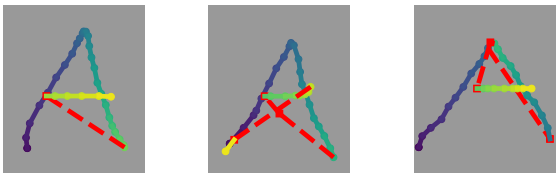


**Fig. 3.** Example of three capital letters $A$ visualizing the movement while writing them

Figure 3 shows three letters interpreted as movement. The dots in the figure represent the start and end points of the movement vectors. The colors of the lines visualize the direction of the movement. The sequence starts with dark purple and ends with light yellow. Dashed red lines present movements in which the pen is lifted. Considering the left letter $A$ in the figure, the movement is well recognizable. It starts from the bottom left, goes up, and then to the bottom right. The pen is lifted and finally, a horizontal line from the left to the right is drawn. This description includes much more information than just an image of the letter $A$. This information might be helpful for the classification task. However, the motion vectors are also challenging since the children grasp their written letters visually and do not focus on a clear movement. Especially, if the pen is lifted to redraw parts of a letter, this might be challenging to classify. Considering the middle letter in Figure 3, it can be observed that after the letter was finished, the child extended the bottom left part of the capital $A$. Visually, this makes sense and makes it easier to recognize the letter but the motion vectors become more complex due to this extension. The same applies to the right letter of the figure where the top center part of the letter is reworked.

As the movement perspective delivers a chronological sequence, a Recurrent Neural Network (RNN) [10] is well suited to classify the data. In the field of RNNs, LSTMs are very powerful [6]. In contrast to classical RNNs, LSTMs include an additional short-time memory and due to their special structure, they can keep values in their short-time memory for a long time [9]. For this reason, they are called Long Short-Time Memory Networks. Our approach to applying an LSTM is very straight-forward. It is shown in Figure 4. The data is interpreted

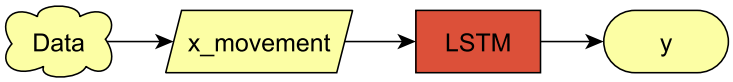



**Fig. 4.** Structure of our deep movement approach

from the movement perspective resulting in $\mathbf{x}_{\text{movement}}$. One pattern $\mathbf{x}_{\text{movement}}$ consists of the 32 movement vectors and the binary information per vector which indicates whether the pen is lifted. The pattern is handed over to the LSTM which classifies it. The classification result is $y$. The employed LSTM includes

1024 cells, followed by a dropout and an output layer. The output layer uses the softmax function. The other layers employ the ReLU function as activation function.

## 3 Multi-Stage Deep Context-Free Handwriting Recognition

In this section, we introduce our new approach. It works context-free, thus, it can be used in educational software as described in Section 1. The idea behind our approach is to combine the advantages of the deep visual and the deep movement approach. For this propose, we employ both approaches and connect their outputs to a Deep Neural Network (DNN) [5] using dense layers. This makes it a multi-stage approach. We denote our multi-stage deep context-free handwriting recognition as C$^-$HR.

*Structure of C$^-$HR* The structure of our approach is shown in Figure 5. Like the structures in the previous section, it is very straight-forward. The figure shows how the deep visual and deep movement approach are combined. In contrast to the approaches employing only one perspective, the outputs of the CNN and LSTM are not used directly. The outputs are handed over to a DNN and the DNN determines the classification result $y$. Thereby, the CNN and LSTM are configured as described in Section 2. As DNN, we employ two dense layers with 1024 neurons per layer. Both layers employ the ReLU function. These layers are followed by a dropout layer and an output layer which uses the softmax activation function.

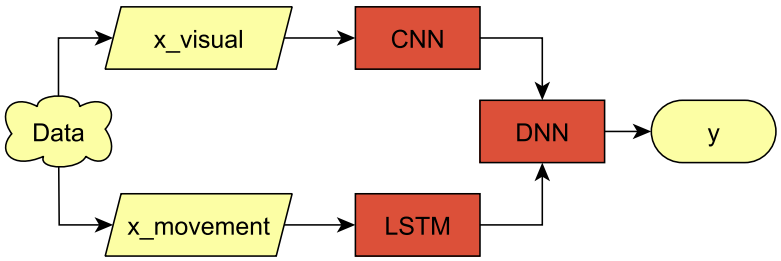



**Fig. 5.** Structure of C$^-$HR

The task of the DNN is to classify a letter based on the information from the CNN and LSTM. To provide the DNN detailed information, the CNN and LSTM are applied without the activation function in their output layer. This is due to the fact that the activation function is the softmax function which scales the output values to a range from 0 to 1. The original range is lost which can offer valuable information. By removing the activation function, this information is kept.

*Benefits of Combining* As described in Section 2, interpreting letters as movement vectors has advantages and disadvantages compared to the visual interpretation of the data. By using both perspectives combined by a DNN, our approach is able to use the strength of both approaches. For example in preliminary experiments, it turned out that the small letters $h$ and $n$ can be better distinguished by employing the visual perspective. This makes sense as the motion for both letters is very similar. In both cases, it is a line down followed by a curve. To distinguish the letters, the length of the line is crucial which can be recognized best by visually examining it. A different example are the small letters $c$ and $l$. They could look very similar if they are written unclearly. Also, the movement is similar but as the upper part of the letter $c$ is a curve and of the letter $l$ is a line, the motion vectors are slightly different. Additionally, the curve is usually written a bit slower than the line. In preliminary experiments, it turned out that the differences between the movement vectors and the writing speed are easier to detect as the visual differences.

## 4 Experimental Results

In this section, the experimental results are presented. For the experiments, we employ three different variants of the data set. The first variant only consists of the capital letters $A$, $M$, $O$, $T$, $U$. We denote this variant as *amotu*. It includes 1473 patterns. In the second variant, all small letters are contained. It consists of 24 810 patterns and we denote it as *small*. The last variant which we call *full* includes all 59 different letters, i.e., all 26 capital and small letters, capital and small special characters (the umlauts: ä, ö, ü), and another special character (eszett: ß). This variant includes 34 627 patterns.

For all experimental results, we employ a 10-point cross-validation using a stratified $k$-folds cross-validator [17]. This provides folds which are preserving the percentage of samples for each class. The neural networks are trained using a keep probability of 0.5 for the dropout layers. They are trained until no more improvements are achieved. Thereby, each component gets the same amount of time. Thus, the deep visual approach which is relatively easy to compute gets more learning epochs. For *amotu*, we train about 270 epochs for the deep visual approach, 80 epochs for the deep movement approach, and 65 epochs for the multi-stage approach. For *small* and *full*, the training time is quintupled. During the training, a batch size of 100 patterns is used. To train C$^-$HR, first, the CNN and LSTM are trained. Then, the DNN which connects the CNN and LSTM is trained. Thereby, the weights of the CNN and LSTM are fixed.

*Comparison of Classification Accuracy* In the first experiment, we compare the classification accuracy of C$^-$HR to the result from [19]. As introduced in Section 1.1, [19] has used a high-level representation to classify the data. Therefore, we denote this approach as HLR. Additionally, we show results of decision trees (DTs) [20] and the more advanced approach of random forest (RF) [3] which is employing multiple DTs. Both approaches are using the visual perspective. We select DTs as their performance can be adjusted by their maximal

depth. This makes it possible to configure a DT with a similar performance to HLR for comparison. The two employed DTs are configured with (1) a maximal depth of 5, leading to similar results as HLR for *amotu*, and (2) none maximal depth, showing the best possible results for a DT. RF is employing 100 DTs with none maximal depth.

**Table 1.** Comparison of the classification accuracy

| Data | HLR | $DT^5$ | $DT^\infty$ | $RF^\infty_{100}$ | $C^-HR$ |
|---|---|---|---|---|---|
| *amotu* | .747 | $.734 \pm .034$ | $.806 \pm .039$ | $.950 \pm .019$ | $\mathbf{.989 \pm .008}$ |
| *small* | - | $.438 \pm .019$ | $.724 \pm .027$ | $.889 \pm .021$ | $\mathbf{.978 \pm .008}$ |
| *full* | - | $.328 \pm .019$ | $.589 \pm .028$ | $.811 \pm .022$ | $\mathbf{.964 \pm .009}$ |

Table 1 shows the results. In the table, the mean accuracies of the 10 folds of the cross-validation and the matching standard deviations are presented. It can be observed that $C^-HR$ clearly outperforms all other approaches. Thereby, it is remarkable that the classification accuracy decreases only a little while increasing the number of letters – from 0.989 for five letters to 0.964 for all 59 letters. The results of the DTs and RF show how the classification accuracy can significantly decrease if the problem is getting more complex due to more letters. Essential is the result of employing all letters. For this task, $C^-HR$ only makes 3.6% prediction errors. Compared to the previous approach, this is a huge improvement.

HLR has only been applied to *amotu* and performs similarly to a DT with a max depth of 5. Obviously, each approach scales differently. However, the results indicate that an approach which is able to predict *amotu* with a classification accuracy of less than 0.75 will likely predict all letter with an accuracy of significantly less than 0.5. Even an approach which is able to achieve a score of 0.95 for *amotu* is just slightly above 0.8 for all letters. Thus, a word with five letters is recognized correctly in less than one-third of the cases ($0.8^5 < 1/3$) by this approach.

*Individual Components of $C^-HR$* Now, we show the results of the components of $C^-HR$, i.e., the performance by the deep visual approach, by the deep movement approach, and by $C^-HR$. This indicates the contributions of the individual components of $C^-HR$. We emphasize this point for two examples at the end of this section.

Table 2 presents the classification accuracies. For *amotu*, the visual approach works very well. This makes sense as the letters *A*, *M*, *O*, *T*, *U* are visually significantly different. For both other data sets, the movement approach performs better than the visual one. This indicates that the additional information due to the movement perspective is beneficial. In all cases, $C^-HR$ is very close to the

**Table 2.** Comparison of the accuracy of the individual components of C$^-$HR

| Data | Visual | Movement | C$^-$HR |
|---|---|---|---|
| *amotu* | $\mathbf{.991 \pm .007}$ | $.986 \pm .011$ | $.989 \pm .008$ |
| *small* | $.962 \pm .011$ | $.975 \pm .007$ | $\mathbf{.978 \pm .008}$ |
| *full* | $.920 \pm .013$ | $.948 \pm .011$ | $\mathbf{.964 \pm .009}$ |

best approach or it is the best. It seems the more complex the data gets, the higher are the benefits from the two perspectives.

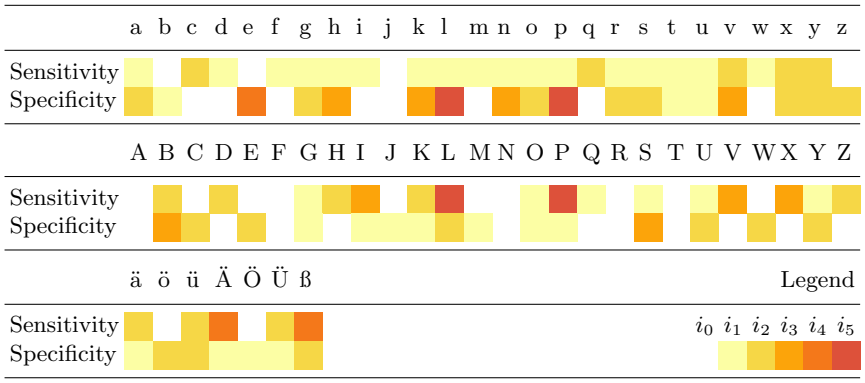
Two examples of letters which benefit from the two perspectives are the small *b* and the small *l*. The small letter *b* is correctly classified by the visual approach with an accuracy of 98.0%. The movement approach only achieves 86.2%. For the small letter *l*, the accuracies are reversed. The visual approach achieves 87.9% while the movement approach classified the letter with an accuracy of 95.5%. To make this more clear, Figure 6 presents four letters which are correctly classified by C$^-$HR. The left two letters are a small *b*. It is correctly labeled using the visual approach. The movement approach classifies both letters as a small *t*. The right two letters are a small *l*. It is also correctly labeled by C$^-$HR The visual approach classifies both letters as a capital *I*.

**Fig. 6.** Example of four letters which are correctly labeled by only one approach

*Sensitivity and Specificity of Letters* To further analyze the performance of our approach, we present the sensitivity and specificity of each letter. For a specific letter, the sensitivity indicates the probability to correctly classify this letter. The specificity indicates the probability to be correctly not classified by applying a different letter. For example, if a small letter *a* is written. A high sensitivity for the letter *a* shows that there is a high chance to correctly predict this letter. A high specificity for the letter *b* indicates that there is a high chance that the written letter is correctly not labeled as a *b*.

Table 3 presents the results. We employ the first fold of the cross-validation. As shown in the legend of the table which is located in the lower right part, there are five different colors from light vanilla to red. Red stands for relatively many errors. The lighter the color gets, the fewer errors are made and if there

**Table 3.** Sensitivity and specificity analysis




is no color, there are no errors. For the sensitivity, the thresholds for the intervals $i_0, \ldots, i_5$ are 80%, 85%, 90%, 95%, 100% and for the specificity, they are 99.80%, 99.85%, 99.90%, 99.95%, 100%. The thresholds for the specificity are much higher as an unknown letter is classified as one letter and it is not classified as the 58 other letters.

All results are close to the thresholds, except the sensitivity of the capital $P$ which is often wrongly classified as a small $p$. Only in 16.1% of the cases, the letter is correctly classified. This makes sense as without using the context, it is very hard to distinguish between the letters $p$ and $P$. This points out that there is potential for improvements, e.g., it might be an approach to add relative height information to our approach. Thereby, we need to keep in mind that our approach stays completely semantic context-free.

To visualize the problems with $p$, Figure 7 shows four letters: on the left side, two capital letters $P$ and the right side, two small letters $p$. All letters are labeled as $p$. The figure indicates that the distinction between the small and capital $p$ is very challenging without additional information. Besides the problems with the capital $P$ which is often classified as the small $p$, the table visualizes other noticeable findings. For example, the sensitivity of the capital $I$ is relatively low, leading to a relatively low specificity of the small $l$. Also, the capital $B$ and the special character $ß$ can be written very similarly. This can be seen in the sensitivity of the special character $ß$ and the specificity of the capital $B$.

Overall, the results of the sensitivity and specificity are satisfactory. Multiple letters are recognized perfectly and a lot of letters are within the first interval. The letters within the last interval point to opportunities to improve our approach.

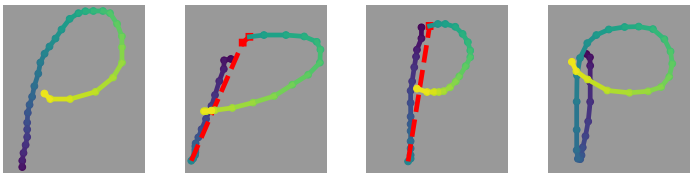


**Fig. 7.** Examples of four small and capital letters *p*

## 5 Conclusions

Handwriting is a crucial skill and is important for success at school. This includes correct orthography. To realize educational software which supports the development of orthography, a context-free handwriting recognition is required. In contrast to usually used recognitions, a context-free handwriting recognition does not imply a correction of misspelled words which makes it possible to detect all writing errors. This is essential for educational software.

In our work, we developed a context-free handwriting recognition which is based on multiple techniques from the field of deep learning. It interprets the data from two perspectives: a visual one and a movement oriented one. The visual perspective is realized as a CNN and the movement perspective as an LSTM. Both perspectives are combined by a DNN. In the experimental results, our new approach can show its superior classification ability compared to the previous approach. It turned out that the visual and movement approach can benefit from each other, especially if the data set contains many letters. Additionally, our results show that it is very challenging to distinguish between capital and small letters in some cases, e.g., in the case of the letter *p*.

In our future work, we are going to tackle the potentials for further improvements, e.g., the implementation of a semantic-free support to better distinguish between small and capital letters. Also, an analysis of the data like in [15] could lead to further findings.

## References


1. Aghdam, H., Heravi, E.: Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification. Springer International Publishing (2017), https://books.google.de/books?id=72UkDwAAQBAJ
2. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer (2007)
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (Oct 2001)
4. Frerichs, J.: Tintenklex Legasthenie Software, Digita-Preisträger 2006, Legasthenie und Dyskalkulie (2018), accessed March 2018: http://www.legasthenie-software.de/
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016)
6. Graves, A., r. Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 6645–6649 (May 2013)

7. Grund, M.: Rechtschreibtraining mit Computer und Lernkartei (2018), accessed March 2018: https://www.gut1.de/
8. Gut, J., Reimann, G., Grob, A.: Kognitive, sprachliche, mathematische und sozial-emotionale Kompetenzen als Prädiktoren späterer schulischer Leistungen. Zeitschrift für Pädagogische Psychologie **26**(3), 213–220 (2012)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (Nov 1997)
10. Jain, L.C., Medsker, L.R.: Recurrent Neural Networks: Design and Applications. CRC Press, Inc., Boca Raton, FL, USA (1999)
11. Jozefowicz, R., Zaremba, W., Sutskever, I.: An empirical exploration of recurrent network architectures. In: Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37. pp. 2342–2350. ICML'15, JMLR.org (2015), http://dl.acm.org/citation.cfm?id=3045118.3045367
12. Lawrence, S., Giles, C.L., Tsoi, A.C., Back, A.D.: Face recognition: a convolutional neural-network approach. IEEE Transactions on Neural Networks **8**(1), 98–113 (Jan 1997)
13. LeCun, Y., Haffner, P., Bottou, L., Bengio, Y.: Object Recognition with Gradient-Based Learning, pp. 319–345. Springer Berlin Heidelberg, Berlin, Heidelberg (1999)
14. Lu, L., Zheng, Y., Carneiro, G., Yang, L.: Deep Learning and Convolutional Neural Networks for Medical Image Computing: Precision Medicine, High Performance and Large-Scale Datasets. Advances in Computer Vision and Pattern Recognition, Springer International Publishing (2017)
15. Lückehe, D., Schmidt, K., Plotz, K., von Voigt, G.: Analysis of sound localization data generated by the extended mainzer kindertisch. In: KI 2017: Advances in Artificial Intelligence - 40th Annual German Conference on AI, Dortmund, Germany, September 25-29, 2017, Proceedings. pp. 330–336 (2017)
16. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. pp. 807–814. ICML'10, Omnipress, USA (2010)
17. Olson, D., Delen, D.: Advanced Data Mining Techniques. Springer Berlin Heidelberg (2008)
18. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: A comprehensive review. Neural Computation **29**(9), 2352–2449 (2017)
19. Reinders, C., Baumann, F., Scheuermann, B., Ehlers, A., Mühlpforte, N., Effenberg, A., Rosenhahn, B.: On-the-fly handwriting recognition using a high-level representation. In: Computer Analysis of Images and Patterns - 16th International Conference, CAIP 2015, Valletta, Malta, September 2-4, 2015 Proceedings, Part I. pp. 1–13 (2015)
20. Rokach, L., Maimon, O.: Data Mining with Decision Trees: Theory and Applications. World Scientific Publishing Company (2014)
21. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**, 1929–1958 (2014)
22. Steinbrink, C., Lachmann, T.: Lese-Rechtschreibstörung: Grundlagen, Diagnostik, Intervention. Springer Berlin Heidelberg (2014)