Report

A phoneme-based student model for adaptive spelling training

Author(s):
Baschera, Gian-Marco; Gross, Markus H.

Publication Date:
2009

Permanent Link:
https://doi.org/10.3929/ethz-a-006733692 →

Rights / License:
In Copyright - Non-Commercial Use Permitted →

ETH Library

A Phoneme-Based Student Model for Adaptive Spelling Training

Gian-Marco BASCHERA and Markus GROSS,

Department of Computer Science, ETH Zürich, Switzerland

Abstract. We present a novel phoneme-based student model for spelling training. Our model is data driven, adapts to the user and provides information for, e.g., optimal word selection. We describe spelling errors using a set of features accounting for phonemic, capitalization, typo, and other error categories. We compute the influence of individual features on the error expectation values based on previous input data using Poisson regression. This enables us to predict error expectation values and to classify errors probabilistically. While our main focus is on spelling training for dyslexic children, our model is generic and can be utilized within any intelligent language learning environment.

Keywords. spelling, student model, phoneme, adaptivity, error classification

Introduction

Intelligent, computer-based language training environments are gaining increasing importance. A key ingredient of such systems is an individualized student model [1], a representation that accounts for student behavior based on background knowledge of the student and the domain. Learning how to spell words is fundamental to language training and poses a great difficulty for students with dyslexia. The focus of our research is the design of an adaptive student model for effective spelling training using machine learning and statistical methods.

A core challenge when building such a model is to identify patterns and similarities in spelling errors across the entire word data base and to represent them using as few parameters as possible. For instance, Bodén [2] proposed a language-specific set of 68 patterns to describe spelling difficulties. This allows for a detection of potential errors based on grapheme groups. In their evolutionary approach of adapting spelling exercises, they essentially select similar words if an error occurs. Error localization and classification with respect to specific difficulties are not considered. Our approach takes these ideas a significant step further. Central features of our design include error localization, the modeling of different error categories and the estimation of their probabilities given erroneous inputs. This allows for a much more refined student model and for a consistent representation of specific spelling difficulties throughout the word data base.

Our new model is data driven and the result of an extensive analysis of a user study [3] that has been carried out to evaluate the Dybuster training software [4]. It includes a multi-modal spelling training for dyslexic children, whose word selection is essentially based on a symbol confusion matrix (SCM).

Our paper follows the three main elements of a student model as proposed by Sison et al. [5]. The section student behavior (1) describes the user data available to the student model. Next, we discuss the background knowledge about spelling errors and characteristic features on a literal as well as on a phonological level (2). The section student model (3) presents the statistical methods used to compute the parameters characterizing the student. Our evaluation shows that the model can classify and predict spelling errors robustly and that its results correspond well with observed error patterns.

1. Student Behavior

The input data for our student model is derived from data gathered during an extensive user study with the Dybuster learning environment in 2006. A group of 80 German-speaking children (43 of which dyslexic) participated in the 6-month training program, including cross-over tests and control groups. All user inputs were stored in logfiles and time-stamped. The writing tests at the beginning and at the end of the study displayed a very high effectiveness of the training. Details can be found in [3].

The analysis of the logfiles with all user inputs revealed the limitations of the employed student model based on SCM. Specifically, SCM, containing $29^2 = 841$ parameters, captures only letter confusions and misinterprets e.g. letter omission (Example: *Spiel* - `Spil`: SCM stores an *'e'-'l'* confusion.). These limitations were the major motivation for our research. In our setting words are prompted orally and have to be typed in by the student using a keyboard. Direct visual and auditory feedback supports the student during training process. A signal tone responds to erroneous input so as to encourage the student to correct the error letter immediately. This immediate correction is paramount to effective training, however, it restricts the error analysis of the input string to the actual error symbol making unambiguous error classification more difficult. We illustrate this with the following example:

Unmut /ʊnmuːt/ - `Unn` The confusion of the letter *'m'* and *'n'* could be due to a doubelling of the *'n'*, due to a confusion of similar phonemes /m/ and /n/, or due to the small key distance of *'m'* and *'n'*, (typo).

This example also shows that some errors are not unambiguously classifiable, even manually. For this reason, we introduce a set of features to characterize errors. Analyzing the available input data using these features enables us to estimate the student's error characteristics and to provide a probabilistic error classification.

2. Background Knowledge and Error Model

Spelling errors of the categories typo and capitalization as well as parts of the letter confusion (see also Table 1) can be modeled by comparing correct and false letters directly. Such errors are correctly represented using a simple symbol confusion matrix. However, our analysis of all misspellings in the user study clearly revealed that most of the errors can be traced back to difficulties on the phonological level. As an example, the afore-described error *Spiel* /ʃpiːl/ - `Spil` is caused by the diversity of grapheme representations (*'i'*, *'ie'*, *'ih'* and *'ieh'*) of the phoneme /iː/. In order to model such phoneme-grapheme based errors we introduce language specific, phoneme-based features.

Table 1. Error categories and their corresponding features as implemented by our model

Category	Features	
Typo: Error committed due to typing difficulties. Strongly dependent on the input device used.	**Key distance** (categorical): *Left/Right*, *Top/Bottom*, *Distant*	letter level
	Technical (binary): Input device specific confusion between umlaut and corresponding vowel.	
Capitalization (Cap): Error due to upper and lower case confusion.	**Capitalization** (categorical): *ToLowerCase*, *ToUpperCase*, *CorrectCase*.	
Letter Confusion (LetC): Confusion of letters can be caused by visual similarity of letters (e.g. *'d'-'b'*) or by auditory similarity of corresponding sounds (/n/-/m/). Both are typical difficulties for dyslexic children.	**Visual Similarity (VS)** (numerical): Based on normalized cross-correlation between images of letters. Computed on actual and horizontally mirrored image for lower, upper and the combination of lower and upper case representations.	
	Auditory Similarity (AS) (categorical): Hierarchical phoneme structure (Section 2.1)	phoneme level
Phoneme Omission (PhoO): Error of leaving out an entire phoneme representation.	**Phoneme Omission** (binary): Phoneme alignment (Section 2.2)	
Phoneme-Grapheme Matching (PGM): Entering wrong representation of correct phoneme. These errors are caused by the non-bijectivity of the phoneme-grapheme correspondence.	**Phoneme Matching (PM)** (categorical): Phoneme alignment (PhoA) (Section 2.2)	
	Elongation (El) (categorical): PhoA (Sec. 2.2)	
	Sharpening (Sh) (categorical): PhoA (Sec. 2.2)	
Phoneme Insertion & Phoneme Transposition: Insertion of an entire phoneme and transposition of two phonemes in a word.	The detection of both categories require information about the complete user input. Therefore, these categories can not reliably be detected.	

Table 1 summarizes all categories of errors a student can make during word spelling training as well as the corresponding features to detect them. A detailed discussion of all features is beyond the scope of this paper. We will therefore focus on the novel phoneme-based features for auditory similarity, phoneme omission and phoneme-grapheme matching, which will be described in detail in Sections 2.1 and 2.2.

2.1. Auditory Similarity (AS)

Our auditory similarity measure between a correct and a false phoneme is based on the hierarchical phoneme structure proposed by Dekel et al. [6]. We modified the structuring of vowels to better address our findings regarding vowel confusion probabilities in the user data (see Figure 1). Confusions between nearby phonemes along the edges /i/-/a/ and /a/-/u/ of the so-called vowel triangle [7] are more likely to happen and thus labeled as similar. We define auditory similarity (AS) as a categorical feature representing the nearest common ancestor node of the correct and the false phoneme.

2.2. Phoneme Alignment

To detect the phoneme omission (PhoO) and phoneme-grapheme matching (PGM) errors, we locally align the user input and the phonological structure of the correct word. We then test the false letter against the current, the following, and the previous phoneme:

PhonemeMatching A false letter can be part of a wrong representation of the correct phoneme. E.g. in Figure 2.a) the false letter *'e'* is the beginning of the grapheme *'eu'*, which is a representative of the correct phoneme /ɔy/.
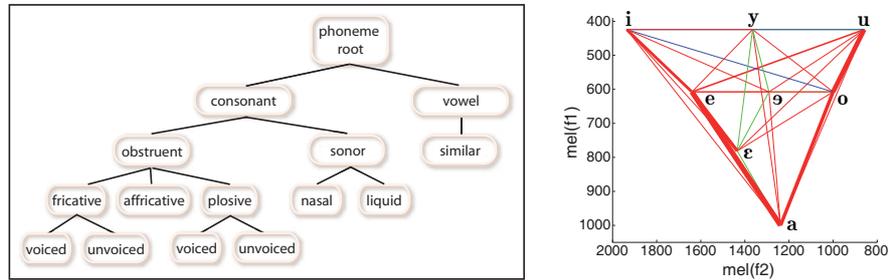
Figure 1. Hierarchical phoneme structure *(left)*: Each phoneme is assigned to one of the leaf nodes, each representing a phonetic attribute. Phoneme triangle *(right)*: Vowels are positioned with respect to their first and second formant frequencies. Empirical confusion probabilities are represented by the thickness of the connecting red lines. Blue and green lines indicate a significant influence of other features (key distance and technical).

Phoneme- and LetterOmission   If a false letter marks the beginning of the next phoneme and the current phoneme is completely omitted, we detect a *PhonemeOmission*. As displayed in Figure 2.b), the incorrectly entered grapheme *'r'* matches the following phoneme /ɐ̯/. The current phoneme /ɛ/ has been omitted. However, if the current phoneme is not omitted, but falsely represented by the previous input grapheme, we face a *LetterOmission*, such as in Figure 2.c). Here, the error letter *'l'* matches the following phoneme, and the current phoneme /iː/ is incorrectly represented by the grapheme *'i'*.

LetterAddition   The previous input grapheme together with the false letter can match the previous phoneme. In Figure 2.d) the false letter *'h'* appended to the previous input grapheme *'a'* results in the grapheme *'ah'* - which is a representative of the previous phoneme /aː/.

To discriminate the error in greater detail, we further subdivide the categories from above. In *PhonemeMatching*, we distinguish between *Vowel* and *Consonant* phonemes as well as between *Main* and *Special* graphemes. The attributes *Main* and *Special* are manually attached to every grapheme. They indicate whether a grapheme is the most likely (main) representative of the phoneme or an unusual (special) one. *LetterOmission* and *LetterAddition* are both subdivided into *Elongation* and *Sharpening* based on the type of phoneme the error occurred in (*Vowel/Consonant*). These features are language specific and the phoneme-grapheme correspondence has to be adapted for each language.
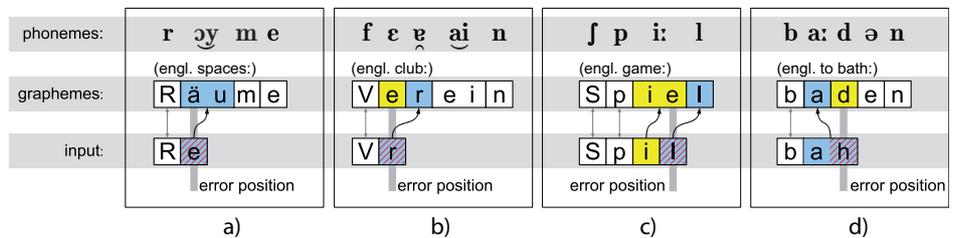

Figure 2. Alignment of correct and input phonemes and resulting error categories: a) Phoneme matching b) Phoneme omission c) Letter omission d) Letter addition

3. Student Model

The presented features characterize isolated errors. By analyzing the available input data of a student, our model estimates the particular difficulties a student has on the types of error described by each feature. To enable a regression analysis of the user data, we replace all categorical variables by $n-1$ dummy variables, where n is the number of categories of the original variable. This results in a $K = 32$ dimensional feature vector $\mathbf{f}$ describing a specific error. In order to allow for an adaptation to an individual student our model computes the following two statistical entities:

$\mathrm{P}(C = k|V = \mathbf{f}(k_e, k_c, w_c))$ The probability that the kth feature causes the actual error. The error is described by the feature vector $\mathbf{f}$ which depends on the false letter k_e, the correct letter k_c and the correct word w_c. This error probability is essential for a local adaptation to an erroneous input. (From now on denoted as $\mathrm{P}_C(k|\mathbf{f})$).

$\mathrm{E}\left[E|V = \mathbf{f}(k_e, k_c, w_c)\right]$ The expected number of such errors a student will make, described by an error feature vector defined as above. This information allows for a global adaptation to the student's characteristics. (From now on $\mathrm{E}\left[E|\mathbf{f}\right]$).

3.1. Data Acquisition

The collected user data provides information about the number of times a potential error defined by a feature vector $\mathbf{f}$ occurs ($N(\mathbf{f})$) and how often this error was actually made by the student ($Y(\mathbf{f})$). To compute these numbers, we process all available inputs of a student as follows:

1. Swap every letter k_c of the prompted, correct word w_c with all other letters k_e. For each swap, compute its feature vector $\mathbf{x_i} = \mathbf{f}(k_e, k_c, w_c)$ and increment the corresponding occurrence counter $N(\mathbf{x_i})$.
2. For every error actually made, compute the feature vector $\mathbf{x_i} = \mathbf{f}(k_e, k_c, w_c)$ and increment the corresponding error counter $Y(\mathbf{x_i})$.

The empirical error expectation values $\mathrm{E}\left[E|\mathbf{x_i}\right]$ can in principle be computed naively by $Y(\mathbf{x_i})/N(\mathbf{x_i})$. However, the large number of different feature vectors and their uneven frequency distribution leads to a very slow convergence of the student characteristics. To improve the robustness of our model and to obtain an error classification, we introduce a statistical model to estimate the influence of each feature on the error expectation values.

3.2. Feature Influence

Let $\mathbf{f}^{(\mathbf{k})}(k_e, k_c, w_c)$ be the feature vector $\mathbf{f}(k_e, k_c, w_c)$ with all but the kth feature f_k set to zero. Our model assumes that the influence $\mathrm{E}\left[E|\mathbf{f}^{(\mathbf{k})}\right]$ of the kth feature on the expected number of errors is independent of all other features. Hence

$$\mathrm{E}\left[E|\mathbf{f}\right] = \sum_{k=0}^{K} \mathrm{E}\left[E|\mathbf{f}^{(\mathbf{k})}\right] \tag{1}$$

Let $\mathbf{F}^{(\mathbf{k})}$ by a feature vector with the kth feature set to 1 and all others equal to 0. The influence of the kth feature of $\mathbf{f}$ on the expected number of errors $\mathrm{E}\left[E|\mathbf{f}^{(\mathbf{k})}\right]$ can be expressed as

$$\mathrm{E}\left[E|\mathbf{f^{(k)}}\right] = \mathrm{E}\left[E|\mathbf{F^{(k)}}\right] \cdot f_k = \beta_k f_k \tag{2}$$

where β_k is the parameter describing the normalized influence of the kth feature on the expectation value of the error. We finally compute the error probability and expectation value from above more robustly as

$$\mathrm{P}_C(k|\mathbf{f}) = \frac{\mathrm{E}\left[E|\mathbf{f^{(k)}}\right]}{\mathrm{E}\left[E|\mathbf{f}\right]} = \frac{\beta_k f_k}{\boldsymbol{\beta}\mathbf{f}} \qquad \mathrm{E}\left[E|\mathbf{f}\right] = \sum_{k=0}^{K} \mathrm{E}\left[E|\mathbf{f^{(k)}}\right] = \boldsymbol{\beta}\mathbf{f} \tag{3}$$

To estimate the student parameters $\boldsymbol{\beta}$ from our data, we utilize Poisson regression [8], summarized as follows: In a Poisson distribution, the probability distribution for every variable Y_i ($i = 1, \ldots, M$) is defined as:

$$\mathrm{P}(Y_i) = \frac{e^{-\mu(\mathbf{x_i})N_i}(\mu(\mathbf{x_i})N_i)^{Y_i}}{Y_i!} \tag{4}$$

where $\mu > 0$ denotes the rate parameter and N the number of exposure to risk. Due to the independence assumption on our features, we choose a linear link function $\mu(\mathbf{x_i}) = \boldsymbol{\beta}\mathbf{x_i}$. To finally compute the required parameter estimate $\hat{\boldsymbol{\beta}}$, we employ the maximum likelihood method [8]. The likelihood principle selects the $\hat{\boldsymbol{\beta}}$ which maximizes the log-likelihood function

$$\mathcal{L}(\boldsymbol{\beta}) = \sum_{i=1}^{M} -\boldsymbol{\beta}\mathbf{x_i}N_i + Y_i \log(\boldsymbol{\beta}\mathbf{x_i}N_i) - \log(Y_i!) \tag{5}$$

subject to $\boldsymbol{\beta}\mathbf{x_i} > 0$. The constraint maximization of $\mathcal{L}(\boldsymbol{\beta})$ can be accomplished by an active set algorithm based on sequential quadratic programming (SQL) [9].

4. Results

We evaluated our student model on the user data gathered in the aforedescribed study. Figure 3 displays the estimated student parameters for three subjects. Subject 1 shows dyslexia typical difficulties on visual and auditory similarities and capitalization. Subject 3 rather suffers from weaknesses in the PGM category. The significance of all features has been evaluated using the likelihood ratio (LR) test [8]. The feature *AS(Fluid)* having the lowest LR value of 13.2 is still significant considering the χ^2 value 3.8 for a significance level of $\alpha = 0.05$.

4.1. Error Classification and Estimation of Error Expectation Value

The classification of errors with one dominant feature activated is consistent across all students. For instance, the error *Spiel* - `Spil` is classified as an elongation (PGM) error for all children with over 99% probability. However, the classification of the error *Unmut* - `Unn` varies for the three different subjects (see Figure 4.a)). It is classified as LetC (*AS(Nasal)*), Typo (*Left/Right*), and PGM (*Sh(Addition)*) respectively for subject 1, 2 and 3.
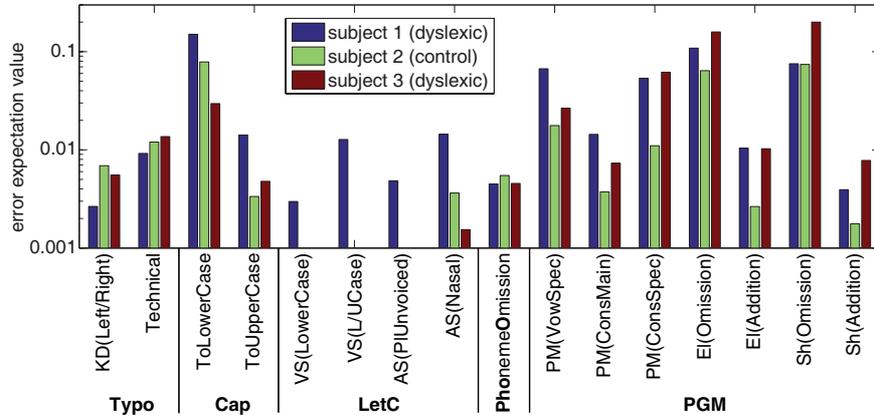
Figure 3. All estimated student parameters β with a value greater than 0.002 for at least one of the three subjects, on a logarithmic scale. Each parameter β_{k} indicates the normalized influence of the kth feature on the error expectation value.

The error expectation values for the word *Männer* are shown in Figure 4.b) for each letter. We can see that subject 1 will make a capitalization error with high probability, while subject 3 will rather likely commit a PGM error at double consonant *'n'*. Such errors are typical for dyslexic students. These results demonstrate the ability of our model to discriminate error types between individual students.

4.2. Verification

The lack of ground truth information makes rigorous verification of the computed error classifications difficult. Therefore, we focus on the student's behavior for error repetition. We assume, for example, that the error category *Typo* is randomly distributed and thus not being conditioned by the time-dependent learning and forgetting process. As a result, the time span between a typo and reselection of the word by the controller should not influence the error repetition probability (ERP), i.e. the probability the student makes the same error again. Conversely, errors of the PGM category, e.g., indicate difficulties of the student with spelling and require training to be remedied. Hence, the longer the time between such errors and the reselection of the word, the higher the ERP. In Figure 4.c)
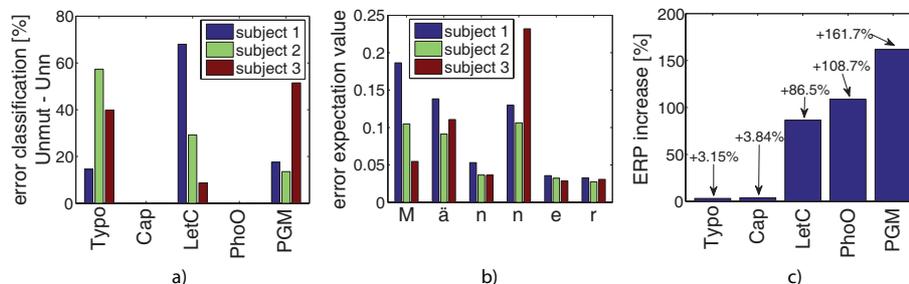

Figure 4. a) Error classification of *Unmut* - `Unn` (engl. resentment). b) Error expectation value for *Männer* (engl. men). c) ERP increase from less than 60s to more than 60s between error and repetition.
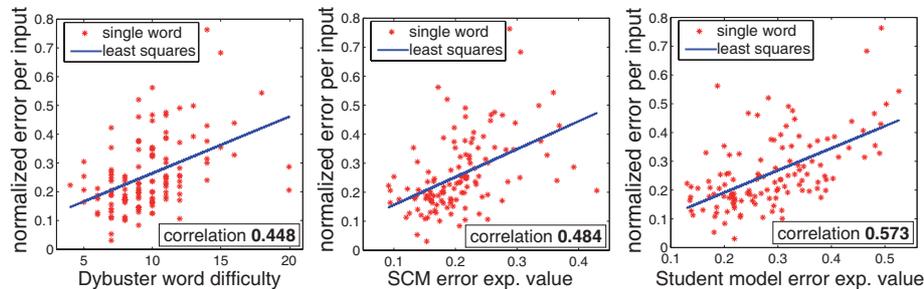
Figure 5. Empirical error expectation values plotted against three word difficulty measures.

we present the increase of the ERP as a function of the time span for word reselection. The ERP of the categories LetC, PhoO and PGM increase significantly, whereas the probability for Typo and Cap stay constant. This corresponds very well with observations made in general spelling experiments with students.

For the verification of error expectation values we compare the correlation of estimated error expectation and empirical (observed) errors per input for three different word difficulty measures computed over the input of all subjects. Figure 5 shows that our difficulty estimation clearly outperforms the student-independent difficulty computation [4] used in Dybuster, as well as the error expectation based on a symbol confusion matrix.

5. Conclusion and Further Work

We introduced a novel adaptive student model to characterize, classify, and predict spelling errors. Our model is data-driven, utilizes statistical methods, and adapts to individual strengths and weaknesses of students. The computed estimates correspond well with experimental observations and with intuition. The model is useful for a wide range of interactive language training. Future work is focused on optimal word selection, aging and forgetting, progress and other temporal aspects of human language acquisition.

References

[1] J. E. Greer and G. I. McCalla (Eds.). *Student Modeling: The Key to Individualized Knowledge-Based Instruction*. Springer Verlag, 1994, 3-540-57510-3.

[2] M. Bodén and M. Bodén. Evolving spelling exercises to suit individual student needs. *Applied Soft Computing*, **7** 2007, 126–135.

[3] M. Kast, M. Meyer, C. Vögeli, M. Gross and L. Jäncke. Computer-based multisensory learning in children with developmental dyslexia. *Restorative Neurology and Neuroscience*, **25** 2007, 355–369.

[4] M. Gross and C. Voegeli. A multimedia framework for effective language training. *Computer & Graphics*, **31** 2007, 761–777.

[5] R. Sison and M. Shimura. Student Modeling and Machine Learning. *International Journal of Artificial Intelligence in Education*, **9** 1999, 128–158.

[6] O. Dekel, J. Keshet and Y. Singer. An Online Algorithm for Hierarchical Phoneme Classification. *Lecture Notes in Computer Science*, 2005, 146–158.

[7] T. A. Hall. *Phonologie: Eine Einführung*. Gruyter, 2000, 3-110-15641-5.

[8] A. C. Cameron and P. K. Trivedi. *Regression Analysis of Count Data*. Cambridge University Press, 1998, 0-521-63567-5.

[9] P. T. Boggs and J. W. Tolle. Sequential Quadratic Programming. *Acta Numerica*, 1995, 1–51.