Poor Man's OCR Post-Correction: Unsupervised Recognition of Variant Spelling Applied to a Multilingual Document Collection

Poor Man's OCR Post-Correction

Harald Hammarström
Department of Linguistics and Philology
Uppsala University
Sweden
harald.hammarstrom@gmail.com

Shafqat Virk
Språkbanken
Gothenburg University
Sweden
shafqat.virk@svenska.gu.se

Markus Forsberg
Språkbanken
Gothenburg University
Sweden
markus.forsberg@svenska.gu.se

ABSTRACT

Simpel OCR post-correction, or more generally,
requires no dictionary
linear in the number
bi
is applied to TODO

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Filtering; H.3.6 [**Information Search and Retrieval**]: Library Automation - Large text archives; I.2.7 [**Computing Methodologies**]: [Artificial Intelligence - Natural language processing]; I.5.4 [**Pattern Recognition**]: [Applications - Text processing]

General Terms

OCR, Multilingual

Keywords

OCR, Multilingual, Unsupervised

1. INTRODUCTION

In search engines and digital libraries, more and more documents become available from scanning of legacy print collections rendered searchable through optical character recognition (OCR). Access to these texts is often not satisfactory due to the mediocre performance of OCR on historical texts and imperfectly scanned or physically deteriorated originals [18, 20]. Historical texts also often contain more spelling variation which turns out to be a related obstacle.

Given this need, techniques for automatically improving OCR output have been around for decades. Most, if not all, of these techniques require resources in the form of a dictionary, a morphological analyzer, annotated training data, (re-)tailoring to a specific language, or the fine-tuning of thresholds or other parameters. However, for many practical applications, especially when it comes to multilingual and/or genre-specific text collections, such resources are expensive to produce or cannot be obtained at all. For these applications, a slim, language-independent, off-the-shelf solution is more attractive and this motivates the present approach.

Given only raw-text data, distributional similarity (e.g., via Word2Vec) as well as form similarity (e.g., edit distance) between types can be computed efficiently. The present paper presents an OCR post-correction algorithm based simply on juxtaposing these two measures: two words are variants of each other if their distributional similarity exceeds that expected by chance from their form similairity. The algorithm uses no language-specific information, runs in linear time and requires no thresholds or parameters to be set (unless parameters are used to obtain the form and/or distributional similarity).

TODO experiment data + evaluated

2. RELATED WORK

Algorithms applicable to the OCR post-correction problem were developed as early as [4], though the target was the similar problem of spelling correction rather than OCR correction per se. A concise survey of trends in subsequent work can be found in [13, 6-13] [14, 1347-1348] and need not be repeated here; instead we summarize the current state-of-the-art. OCR post-correction systems suggest corrections based on form similarity to a more frequent word, and if a dictionary of correct forms is available, positive use can be made of it [6]. Pairs of words with a given edit distance can be found in an efficient, sub-quadratic, running time [16]. A number of features based on form, frequency and dictionary properties may be used to rank candidates [7]. Most systems use labeled training data and treat the task of ranking candidates as a standard supervised machine learning problem [14, 11] though [1, 2] treat it as an SMT problem and thereby make some use of context. A few systems use context explicitly [19] and the work in the present paper can be said to continue this line. However, none the systems so far

DATeCH 2007 Göttingen, Germany

described can be run off-the-shelf; some resources or human interaction is required to produce corrected OCR. The systems covered in [1, 2, 14, 11, 6] necessitate language-specific labeled training data, while [8, 9] need a dictionary, and [3] relies on google to provide a dictionary. The approaches by [7, 17, 19] need some human intervention to set dataset-specific thresholds or some additional component to choose among alternatives. [10] requires several independent OCR engines and alignment of their output.

TODO No standardized dataset

3. POOR MAN'S OCR POST-CORRECTION

We start from raw text data as input.

1. From raw text data we can derive a measure of *distributional similarity*, here denoted $Sim(x, y)$ between terms x, y using a small size window in (now standard) techniques such as Latent Semantic Indexing (LSI) [5] or Word2Vec [12]. The OCR-correction strategy depends very little on the choice of measure for distributional similarity[1].

2. We define a form-neighbourhood function $N(x)$ that gives the set of forms close to x. The exact choice of neighbourhood function depends on ambition, but the natural choice for is the set of forms with edit distance ≤ 1 to x: $N(x) = \{y|ED(x, y) \leq 1\}$.

3. Now we may define the key criterion $V(x, y)$ which assesses whether y is an OCR variant (or, more generally, spelling/string variant) of x. Fixing on a specific x let $Sim_x(y) = S(x, y)$ be the list of similarity values x has to all other terms and let $Sim_x(y)[k]$ denote the kth highest value. $V(x, y)$ is rendered true iff $S(x, y)$ exceeds that expected by chance from $|N(y)|$ trials from $Sim_x(y)$. In other words, suppose we selected $|N(y)|$ values randomly from $Sim_x(y)$ (the similarity x has to all other terms), if the actual similarity $S(x, y)$ of y to x exceeds all of those, then (and only then) y is deemed an OCR variant of x. The expectation when choosing k items from a list of n total items is that the maximum of the k values is at the $k + 1$th quantile. For our purposes then, we need to check if $S(x, y)$ is in the $|N(y)| + 1$th quantile of $Sim_x(y)$. We thus define $V(x, y) = S(x, y) > Sim_x(y)[\frac{|N(y)|}{|N(y)|+1} \cdot |Sim_x(y)|]$.

4. Finding all variants for all terms in the input can now simply be done by computing $V(x, y)$ for all $x, y \in N(x)$. Each outcome equivalence class of variants is then normalized ("corrected") to its most frequent term.

The procedure may be iterated, whereby variants identified in the OCR post-correction are conflated and re-fed to the distributional similarity calculation, which in turn may suggest new OCR variants, until convergence (TODO cf.). Especially if the neighbourhood function is conservatively bound to edit distance 1, iteration is the only way to achieve OCR postcorrection involving more than one character per term.

We will now illustrate the procedure with an example from the dataset used for experiments (see below for details).

[1]We have used the efficient and easily-accessible implementations of LSI and Word2Vec in [15].

1. We run Word2Vec with the default settings[2] to get a vector space representation for each term. As an exmaple, the ten terms most similar to 'language' is shown below.

Rank	y	$S('language', y)$
1	languages	0.7619
2	linguistic	0.7555
3	dialect	0.7381
4	community	0.7074
5	history	0.7036
6	culture	0.6995
7	society	0.6704
8	population	0.6636
9	lexicon	0.6542
10	literature	0.6482

2. As a neighbourhood function, we choose all the one-character substitutions with letters from the English lowercase alphabet (Σ). Consider then the word 'language', $N(language) = aanguage, banguage, \ldots, language, lbng$ contains $|language| \cdot \Sigma 8 \cdot 26 = 208$ forms.

3. Which of these 208 forms have a higher than expected distributional similarity $S(language, y)$ to 'language'? The total vocabulary size is 204 002 and on $|N(language)| = 208$ trials the expected quantile to beat is the $\frac{1}{209} \cdot 204002 \approx 976$th quantile. The 976th highest value of $Sim_{language}(y)$, i.e., $Sim_{language}(y)[976]$ is 0.3839. 7 of the members of $N(language)$ (apart from the term 'language' itself) have a distributional similarity to 'language' higher than this:

y	$Sim(language, y)$	frequency of y
ianguage	0.5346	387
languagc	0.4644	225
languaqe	0.4229	93
languuge	0.4187	68
languago	0.4077	135
lauguage	0.4003	77
lunguage	0.3831	63

These are thus deemed OCR errors to be corrected to 'language' whose frequency (581 815) is far higher.

4. EXPERIMENTS

Dataset spans

5. DISCUSSION

r["cat"] 'eat'

cat vs can

and is thus bounded by the $O(|X| \cdot k)$ where k depends not on |X| but on the length.

Fromanalgorithmicpointofview,ourcomputationof profiles canbeconsideredasaspecialformofanexpectation maximization-procedurewhereglobalandlocalprofilesare improvedusinganit-erativemutualreinforcementprinciple. To guaranteeefficien-cyofthecomputation, specialfinite-state technology hasbeen-developed [12].

[2]For the record, the default settings are CBOW with mean training method, vector dimensionality 100, learning rate 0.025, word window size 5, minimal frequency 5, threshold for random downsampling of higher-frequency words 1e-3, number of noise words word for negative sampling 5, number of epochs 5, batch size 10000.

Tothebestofourknowledgethe first systemdirectedtowardsunsupervisedcorrectingspelling correctionsystempresentedbyChurchandGale [19]. Inthe beginning, alleditoperationsareassumedtobeequallyprobable. From alargetextcorpusallstringsareretrievedthatarenotin the backgroundlexiconandthatleadtoavalidwordwithone edit operation.Iterativelythespellingsystemrunsoverthestring collection andthenusesthecorrectionsmadetoupdatetheedit probabilities heldincharacterconfusionmatrices.

6. CONCLUSION

The *proceedings* are the records of a conference. ACM seeks to give these conference by-products a uniform, high-quality appearance. To do this, ACM has some rigid requirements for the format of the proceedings documents: there is a specified format (balanced double columns), a specified set of fonts (Arial or Helvetica and Times Roman) in certain specified sizes (for instance, 9 point for body copy), a specified live area (18 × 23.5 cm [7" × 9.25"]) centered on the page, specified size of margins (1.9 cm [0.75"]) top, (2.54 cm [1"]) bottom and (1.9 cm [.75"]) left and right; specified column width (8.45 cm [3.33"]) and gutter size (.83 cm [.33"]).

The good news is, with only a handful of manual settings[3], the LaTeX document class file handles all of this for you.

The remainder of this document is concerned with showing, in the context of an "actual" document, the LaTeX commands specifically available for denoting the structure of a proceedings paper, rather than with giving rigorous descriptions or explanations of such commands.

7. THE *BODY* OF THE PAPER

Typically, the body of a paper is organized into a hierarchical structure, with numbered or unnumbered headings for sections, subsections, sub-subsections, and even smaller sections. The command `\section` that precedes this paragraph is part of such a hierarchy.[4] LaTeX handles the numbering and placement of these headings for you, when you use the appropriate heading commands around the titles of the headings. If you want a sub-subsection or smaller part to be unnumbered in your output, simply append an asterisk to the command name. Examples of both numbered and unnumbered headings will appear throughout the balance of this sample document.

Because the entire article is contained in the **document** environment, you can indicate the start of a new paragraph with a blank line in your input file; that is why this sentence forms a separate paragraph.

7.1 Type Changes and *Special* Characters

We have already seen several typeface changes in this sample. You can indicate italicized words or phrases in your text with the command `\textit`; emboldening with the command `\textbf` and typewriter-style (for instance, for computer code) with `\texttt`. But remember, you do not have to indicate typestyle changes when such changes are part of the *structural* elements of your article; for instance, the heading of this subsection will be in a sans serif[5] typeface, but that is handled by the document class file. Take care with the use of[6] the curly braces in typeface changes; they mark the beginning and end of the text that is to be in the different typeface.

You can use whatever symbols, accented characters, or non-English characters you need anywhere in your document; you can find a complete list of what is available in the *LaTeX User's Guide*[**?**].

7.2 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

7.2.1 Inline (In-text) Equations

A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin. . .\end` construction or with the short form `$. . .$`. You can use any of the symbols and structures, from α to ω, available in LaTeX[**?**]; this section will simply show a few examples of in-text equations in context. Notice how this equation: $\lim_{n\rightarrow \infty}x=0$, set here in in-line math style, looks slightly different when set in display style. (See next section).

7.2.2 Display Equations

A numbered display equation – one set off by vertical space from the text and centered horizontally – is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in LaTeX; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n\rightarrow \infty}x=0 \tag{1}$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty}x_i=\int_{0}^{\pi+2} f \tag{2}$$

just to demonstrate LaTeX's able handling of numbering.

7.3 Citations

Citations to articles [**?**, **?**, **?**, **?**], conference proceedings [**?**] or books [**?**, **?**] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation

[3]Two of these, the `\numberofauthors` and `\alignauthor` commands, you have already used; another, `\balancecolumns`, will be used in your very last run of LaTeX to ensure balanced column heights on the last page.

[4]This is the second footnote. It starts a series of three footnotes that add nothing informational, but just give an idea of how footnotes work and look. It is a wordy one, just so you see how a longish one plays out.

[5]A third footnote, here. Let's make this a rather short one to see how it looks.

[6]A fourth, and last, footnote.

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ_1^2	1 in 40,000	Unexplained usage

Figure 1: A sample black and white graphic (.eps format).

commands with a key of the item cited in the proper location in the `.tex` file [?]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the `.bib` file for your article.

The details of the construction of the `.bib` file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *LaTeX User's Guide*[?].

This article shows only the plainest form of the citation command, using `\cite`. This is what is stipulated in the SIGS style specifications. No other citation format is endorsed or supported.

7.4 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper "floating" placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material is found in the *LaTeX User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed dvi output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will "float" to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed dvi output of this document.

7.5 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper "floating" placement of figures, use the environment **figure** to enclose the figure and its caption.

This sample document contains examples of **.eps** and **.ps** files to be displayable with LaTeX. More details on each of these is found in the *Author's Guide*.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper

Figure 2: A sample black and white graphic (.eps format) that has been resized with the `epsfig` command.

"floating" placement of tables, use the environment **figure*** to enclose the figure and its caption. and don't forget to end the environment with figure*, not figure!

Note that either **.ps** or **.eps** formats are used; use the `\epsfig` or `\psfig` commands as appropriate for the different file types.

7.6 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. There are two forms, one produced by the command `\newtheorem` and the other by the command `\newdef`; perhaps the clearest and easiest way to distinguish them is to compare the two in the output of this sample document:

This uses the **theorem** environment, created by the `\newtheorem` command:

THEOREM 1. *Let f be continuous on $[a,b]$. If G is an antiderivative for f on $[a,b]$, then*

$$\int_a^b f(t)dt = G(b) - G(a).$$

The other uses the **definition** environment, created by the `\newdef` command:

Definition 1. If z is irrational, then by e^z we mean the unique number which has logarithm z:

$$\log e^z = z$$

Two lists of constructs that use one of these forms is given in the *Author's Guidelines*.

There is one other similar construct environment, which is already set up for you; i.e. you must *not* use a `\newdef` command to create it: the **proof** environment. Here is a example of its use:

PROOF. Suppose on the contrary there exists a real number L such that

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x\to c} f(x) = \lim_{x\to c} \left[gx \cdot \frac{f(x)}{g(x)} \right] = \lim_{x\to c} g(x) \cdot \lim_{x\to c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that $l \neq 0$. □

Complete rules about using these environments and using the two different creation commands are in the *Author's Guide*; please consult it for more detailed instructions. If you need to use another construct, not listed therein, which you want to have the same formatting as the Theorem or the Definition[?] shown above, use the `\newtheorem` or the `\newdef` command, respectively, to create it.

A *Caveat* for the TeX Expert

Because you have just been given permission to use the `\newdef` command to create a new form, you might think

Table 2: Some Typical Commands

Command	A Number	Comments
`\alignauthor`	100	Author alignment
`\numberofauthors`	200	Author enumeration
`\table`	300	For tables
`\table*`	400	For wider tables

Figure 3: A sample black and white graphic (.eps format) that needs to span two columns of text.

Figure 4: A sample black and white graphic (.ps format) that has been resized with the `psfig` command.

you can use TeX's `\def` to create a new command: *Please refrain from doing this!* Remember that your LaTeX source code is primarily intended to create camera-ready copy, but may be converted to other forms – e.g. HTML. If you inadvertently omit some or all of the `\def`s recompilation will be, to say the least, problematic.

8. CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the LaTeX book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

9. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

10. REFERENCES

[1] H. Afli, Z. Qiu, A. Way, and P. Sheridan. Using SMT for OCR error correction of historical texts. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016.*, pages 962–966, 2016.

[2] H. Afli and A. Way. Integrating optical character recognition and machine translation of historical documents. In *Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH)*, pages 109–116, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[3] Y. Bassil and M. Alwani. OCR post-processing error correction algorithm using google online spelling suggestion. *Journal of Emerging Trends in Computing and Information Sciences*, 3(1):90–99, 2012.

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

[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[6] S. Eger, T. vor der Brück, and A. Mehler. Statistical learning for OCR text correction. *The Prague Bulletin of Mathematical Linguistics*, 105:77–99, 2016.

[7] J. Evershed and K. Fitch. Correcting noisy ocr: Context beats confusion. In *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, DATeCH '14, pages 45–51, New York, NY, USA, 2014. ACM.

[8] K. Kettunen. Keep, change or delete? setting up a low resource ocr post-correction framework for a digitized old finnish newspaper collection. In D. Calvanese, D. De Nart, and C. Tasso, editors, *Digital Libraries on the Move: 11th Italian Research Conference on Digital Libraries, IRCDL 2015, Bolzano, Italy, January 29-30, 2015, Revised Selected Papers*, pages 95–103. Springer International Publishing, Cham, 2016.

[9] I. Kissos and N. Dershowitz. Ocr error correction using character correction and feature-based word classification. In *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, pages 198–203, Los Alamitos, CA, USA, 2016. IEEE Computer Society.

[10] W. B. Lund, E. K. Ringger, and D. D. Walker. How well does multiple ocr error correction generalize? In *Proceedings og SPIE 9021, Document Recognition and Retrieval XXI*, volume 9021, pages 1–13, 2013.

[11] J. Mei, A. Islam, Y. Wu, A. Moh'd, and E. E. Milios. Statistical learning for OCR text correction. *CoRR*, abs/1611.06950, 2016.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013.

[13] K. Niklas. Unsupervised post-correction of ocr errors. Master's thesis, Leibniz Universitĺat Hannover, 2010.

[14] U. Reffle and C. Ringlstetter. Unsupervised profiling of ocred historical documents. *Pattern Recognition*, 46(5):1346–1357, May 2013.

[15] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for*

NLP Frameworks, pages 45–50, Valletta, Malta, May 2010. ELRA.

[16] M. Reynaert. Synergy of nederlab and @philostei: Diachronic and multilingual text-induced corpus clean-up. In N. C. C. Chair), K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1224–1230, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA).

[17] M. Reynaert. OCR post-correction evaluation of early dutch books online - revisited. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016.*, pages 967–974, 2016.

[18] K. Taghva, T. Nartker, and J. Borsack. Information access in the presence of ocr errors. In *Proceedings of the 1st ACM Workshop on Hardcopy Document Processing*, HDP '04, pages 1–8, New York, NY, USA, 2004. ACM.

[19] X. Tong and D. A. Evans. A statistical approach to automatic ocr error correction in context. In *Fourth Workshop on Very Large Corpora*, pages 88–100, 1996.

[20] M. C. Traub, J. van Ossenbruggen, and L. Hardman. Impact analysis of OCR quality on research tasks in digital archives. In *Research and Advanced Technology for Digital Libraries - 19th International Conference on Theory and Practice of Digital Libraries, TPDL 2015, Poznań, Poland, September 14-18, 2015. Proceedings*, pages 252–263, 2015.

APPENDIX

A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e. the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters

A.2.2 Math Equations

Inline (In-text) Equations.

Display Equations.

A.2.3 Citations

A.2.4 Tables

A.2.5 Figures

A.2.6 Theorem-like Constructs

A Caveat for the TEX Expert

A.3 Conclusions

A.4 Acknowledgments

A.5 Additional Authors

This section is inserted by LATEX; you do not insert it. You just add the names and information in the `\additionalauthors` command at the start of the document.