Author Verification by Linguistic Profiling: An Exploration of the Parameter Space

HANS VAN HALTEREN
Radboud University Nijmegen

This article explores the effects of parameter settings in linguistic profiling, a technique in which large numbers of counts of linguistic features are used as a text profile which can then be compared to average profiles for groups of texts. Although the technique proves to be quite effective for authorship verification, with the best overall parameter settings yielding an equal error rate of 3% on a test corpus of student essays, the optimal parameters vary greatly depending on author and evaluation criterion.

Categories and Subject Descriptors: I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; I.5.1 [**Pattern Recognition**]: Models—*Statistical*; I.7.m [**Documents and Text Processing**]: Miscellaneous; J.5 [**Arts and Humanities**]—*Linguistics, literature*

General Terms: Algorithms, Experimentation, Security

Additional Key Words and Phrases: Authorship attribution, authorship recognition, authorship verification, machine learning

ACM Reference Format:

Van Halteren, H. 2007. Author verification by linguistic profiling: An exploration of the parameter space. ACM Trans. Speech Lang. Process. 4, 1, Article 1 (January 2007), 17 pages. DOI = 10.1145/1187415.1187416 http://doi.acm.org/10.1145/1187415.1187416

1. INTRODUCTION

There are several situations in language research or language engineering where we are in need of a specific type of extra-linguistic information about a text (document), and we would like to determine this information on the basis of linguistic properties of the text. Examples are the determination of the language variety or genre of a text, or a classification for document routing or information retrieval. For each of these applications, techniques have been developed focusing on specific aspects of the text often based on frequency counts of function words in linguistics and of content words in language engineering.

Authors' address: Radboud University Nijmegen, The Netherlands; email: hvh@let.ru.nl.

In the technique we are presenting in this article, linguistic profiling, we make no a priori choice for a specific type of word (or more complex feature) to be counted. Instead, all possible features are included, and it is determined by the statistics for the texts under consideration and the distinction to be made, how much weight, if any, each feature is to receive. Furthermore, the frequency counts are not used as absolute values but rather as deviations from a norm, which is again determined by the situation at hand. Our hypothesis is that this technique can bring a useful contribution to all tasks where it is necessary to distinguish one group of texts from another. In this article, the technique is tested for one specific type of group, namely, the group of texts written by the same author.

2. TASKS AND APPLICATION SCENARIOS

Traditionally, work on the attribution of a text to an author is done in one of two environments. The first is that of literary and/or historical research where attribution is sought for a work of unknown origin (e.g., Mosteller and Wallace [1984]; Holmes [1998]). As secondary information generally identifies potential authors, the task is authorship recognition, that is, selection of one author from a set of known authors. There is a lively interest in this task, demonstrated the fact that as many as ten groups participated in the Ad-Hoc Authorship Attribution Contest organized by Patrick Juola and presented during the 2004 ALLC/ACH Conference. The best performing systems all used machine learning approaches, but with different choices of techniques and features used. Interestingly, there were two classes of best performing systems. On abundant data problems (generally full books), Koppel and Schler were the clear winners, using as primary features the choice made between commonly used mutual substitutes [Koppel et al. 2003], backed up by most frequent function words, words and part-of-speech tags [Koppel and Schler 2003]. On sparse data problems (e.g., the student essays used in this article), the field was led by Kešelj [2003], looking at the most frequent byte-N-grams, and by a previous incarnation of linguistic profiling [van Halteren 2004].

The second environment for text attribution is that of forensic linguistics, where it needs to be determined if a suspect did or did not write a specific, probably incriminating, text (e.g., Broeders [2001]; Chaski [2001]). Here the task is authorship verification: confirming or denying authorship by a single known author. We here focus on yet a third environment, namely, the handling of large numbers of student essays.

For some university courses, students have to write one or more essays every week and submit them for grading. Authorship recognition is needed in the case of the sloppy student who forgets to include his name in the essay. If we could link such an essay to the correct student ourselves, it would prevent delays in handling the essay. Authorship verification is needed in the case of the fraudulent student who has decided that copying is much less work than writing an essay himself, which is only easy to spot if the original is also submitted by the original author.

In both scenarios, the test material will be sizable, possibly around a thousand words, and at least several hundred. Training material can be sufficiently

available as well, as long as text collection for each student is started early enough. Many other authorship verification scenarios do not have the luxury of such long stretches of test text. For now, however, we prefer to test the basic viability of linguistic profiling on such longer stretches. Later, further experiments can show how long the test texts need to be to reach an acceptable recognition/verification quality.

2.1 Quality Measures

For recognition, quality is best expressed as the percentage of correct choices when choosing between N authors, where N generally depends on the attribution problem at hand. We will use the percentage of correct choices between two authors in order to be able to compare with previous work. For verification, quality is usually expressed in terms of erroneous decisions. When the system is asked to verify authorship for the actual author of a text and decides that the text was not written by that author, we speak of a false reject. The false reject rate (FRR) is the percentage of cases in which this happens, the percentage taken from the cases that should be accepted. Similarly, the false accept rate (FAR) is the percentage of cases where somebody who has not written the test text is accepted as having written the text. With increasing threshold settings, FAR will go down, while FRR goes up. The behavior of a system can be shown by one of several types of FAR/FRR curves, such as the receiver operating characteristic (ROC). Alternatively, if a single number is preferred, a popular measure is the equal error rate (EER), that is, the threshold value where FAR is equal to FRR. However, the EER may be misleading, since it does not take into account the consequences of the two types of errors. For the example application of plagiarism detection, we would not want to reject, that is, accuse someone of plagiarism, unless we are sure. So we would also like to measure the quality of the system with the false accept rate at the threshold at which the false reject rate becomes zero. Another possible choice is the reverse: false reject rate at the threshold at which the false accept rate becomes zero. This would be useful for the extremely strict judge who does not want any attempt at fraud to succeed. Then, if we are not sure beforehand where we want to set the threshold, the system's average quality could be measured by the surface under the ROC curve. As a final measure, we move from the verification viewpoint to the IR viewpoint and use the popular F-measure, the harmonic mean of precision and recall.

2.2 The Test Corpus

Before using linguistic profiling for any real task, we should test the technique on a benchmark corpus. The first component of the Dutch Authorship Benchmark Corpus (ABC-NL1) appears to be almost ideal for this purpose. It contains widely divergent written texts produced by first-year and fourth-year students of Dutch at the University of Nijmegen. The ABC-NL1 consists of 72 Dutch texts by 8 authors, controlled for age and educational level of the authors, and for register, genre, and topic of the texts. It is assumed that the authors' language skills were advanced but their writing styles were weakly developed, and hence very similar, unlike those in literary attribution problems.

Each author was asked to write nine texts of about a page and a half. In the end, it turned out that some authors were more productive than others and that the text lengths varied from 628 to 1342 words. The authors did not know that the texts were to be used for authorship attribution studies, but instead assumed that their writing skill was being measured. The topics for the nine texts were fixed so that each author produced three argumentative nonfiction texts on the television program Big Brother, the unification of Europe, and smoking, three descriptive nonfiction texts about soccer, the (then) upcoming new millennium, and the most recent book they read, and three fiction texts, namely, a fairy tale about Little Red Riding Hood, a murder story at the university, and a chivalry romance.

The ABC-NL1 corpus is not only well-suited because of its contents, it has also been used in previously published studies into authorship attribution. A traditional authorship attribution method, that is, using the overall relative frequencies of the fifty most frequent function words and a principal components analysis (PCA) on the correlation matrix of the corresponding 50-dimensional vectors, fails completely [Baayen et al. 2002]. The use of linear discriminant analysis (LDA) on overall frequency vectors for the 50 most frequent words achieves around 60% correct attributions when choosing between two authors, which can be increased to around 80% by the application of cross-sample entropy weighting [Baayen et al. 2002]. Weighted Probability Distribution Voting (WPDV) modeling on the basis of a very large number of features achieves 97.8% correct attributions [van Halteren et al. 2005]. Finally, an earlier instantiation of linguistic profiling achieves a 99.4% score on this task [van Halteren 2004]. Although designed to produce a hard recognition task, the latter results show that very high recognition quality is feasible. Still this appears to be a good test corpus to examine the effectiveness of any proposed technique.

3. LINGUISTIC PROFILING

In linguistic profiling, the occurrences of a large number of linguistic features in a text, either individual items or combinations of items, are counted. These counts are then normalized for text length, and it is determined how much (i.e., how many standard deviations) they differ from the mean observed in a profile reference corpus. For the authorship task, the profile reference corpus consists of the collection of all attributed and nonattributed texts, that is, the entire ABC-NL1 corpus. For each text, the deviation scores are combined into a text profile vector V_T, on which a variety of distance measures can be used to position the text in relation to any group of other texts.

3.1 Features

Many types of linguistic features can be profiled, such as features referring to vocabulary, lexical patterns, syntax, semantics, pragmatics, information content, or item distribution through a text. However, we decided to restrict the current experiments to a few simpler types of features to demonstrate the overall techniques and methodology for profiling before including every possible type of feature. In this article, we focus on lexical and syntactic features.

3.2 Experimental Setup

In the problem at hand, the system has to decide if an unattributed text is written by a specific author on the basis of attributed texts by that and other authors. We test our system's ability to make this distinction by means of a 9-fold cross-validation experiment. In each set of runs of the system, the training data consists of attributed texts for eight of the nine essay topics. The test data consists of the unattributed texts for the ninth essay topic. This means that, for all runs, the test data is not included in the training data and is about a different topic than what is present in the training material. During each run within a set, the system only receives information about whether each training text is written by one specific author. All other texts are only marked as "not by this author".

3.3 Authorship Score Calculation

The system first builds an author profile vector V_A to represent text written by the author in question. This is simply the featurewise average of the profile vectors V_T of all text samples marked as being written by the author in question. The system then determines a raw score for all text samples in the list, calculated as the difference between the distance from the text profile vector to the author profile vector and the distance from the text profile vector to the zero profile vector:

$$ScoreA(T) = \Delta A(VT, 0) - \Delta A(VT, VA).$$

By subtracting the distance $\Delta_A(V_T, V_A)$, the score will grow with the similarity between the text profile and the author profile. The positive component serves as a correction factor for the length of the text profile vector. Note that this correction factor is an inheritance from the previous empirically-inspired version of Linguistic Profiling and that its exact form also merits some closer investigation in future work.

3.4 Distance Measure

Rather than using the normal distance measure, we opted for a nonsymmetric measure which includes three weighting functions. For each individual vector position (feature) i, we calculate three factors.

Feature difference.

$$Diff_i = W_D(|T_i - A_i|),$$

with T_i the value for feature i in V_T, A_i the value for feature i in V_A, and W_D a weighting function which will be described later.

Feature importance for this text.

$$TImp_i = W_T(max(|T_i| - 1, 0)).$$

With this factor, we model how atypical a specific relative feature count in a text is. Since all feature scores represent the number of standard deviations that the current count is away from the average, a score between -1 and 1 is

unremarkable. So if we subtract 1 from the feature score and then round up to zero for negative results, we introduce a base importance factor which is zero within the −1 to 1 range and starts growing outside this range. How fast it grows is determined by the weighting function W_T, which is again explained later.

Feature importance for this author.

$$AImp_i = W_A(max(|A_i| - STD(i), 0))$$

With this factor, we model how typical or atypical the counts for a specific feature are in the texts of this author. The component STD(i) represents the standard deviation for the values for feature i in all texts. So the base importance value $|A_i| - STD(i)$, rounded up to zero if negative, indicates whether the current author is on average in the expected range, with a value of zero when at most one standard deviation away from the norm and one when two standard deviations away. The further the author's average is away from the norm, the more weight we want to give to this feature. How much more weight is given to this feature is determined by a weighting function W_A, which is explained in the following.

The two importance factors are then multiplied per feature:

$$Imp_i = AImp_i TImp_i,$$

and the overall distance is calculated as

$$\Delta_A = W_D^{-1}(\Sigma_i(Diff_i Imp_i)/\Sigma_i(Imp_i)).$$

This final operation is meant to make the results for different distance calculations more comparable.

3.5 Weighting Function Parameters

In the previous section, each factor in the distance calculation was adjusted with a weighting function. In the investigations in this article, we consider two types of functions for each of W_D, W_A, and W_T. They can either be polynomial x^C or exponential C^x, with C in both cases the weighting constant. This means we have six parameters in all, that is, the (binary) choice of function type for each of the three functions and the (real-valued) constant C for each of the three functions.

In our experiments, we have varied polynomial C from 0.2 to 2.4 by +0.2 for W_D. We did not consider negative or zero values since we deemed the difference to be a clearly positive factor. In addition, we used an exponential C from ½ to 128 by ×2. For W_A, the polarity was less clear, since the hypothesis about what is useful is less strong. Still we focused on the positive range, with polynomial C varying from −1.0 to +3.0 by +0.5. Again, we used an exponential C from ½ to 128 by ×2. For W_T, earlier experiments had shown lower importance (and hence we actually ignored this factor in earlier papers). In this investigation, we decided to examine some variation, with polynomial C varying from −1.0 to +1.0 by +0.5 and exponential C from ½ to 2 by $\times\sqrt{2}$. We will see in the

following how the effectiveness of the measure varies with the setting of the parameters.

3.6 Normalization

The order of magnitude of the raw score values obviously varies with the parameter settings. Furthermore, the values can fluctuate significantly with the sample collection. In the previous work on linguistic profiling, we brought the values into a range which is suitable for subsequent calculations by expressing them as the number of standard deviations they differ from the mean of the scores of the text samples marked as not being written by the author in question. In this article, we decided to view this operation as a further parameter setting, namely, use normalization (relative values: REL) or not (absolute values: ABS).

3.7 Threshold

There is in fact one further parameter, namely, the threshold above which we deem a text to be a positive hit. In this article our main evaluation criteria often abstract from this threshold, since they are based on the ROC curve which considers all thresholds at once. The author recognition accuracy also does not need threshold settings as it deals with relative rather than absolute values. Only the F-measure needs a threshold parameter, but even here we will ignore it and instead report on the optimal value of the F-measure. This means that, in the main part of this article, the parameter status of the threshold is ignored. However, when we investigate trainability of parameter settings, we do include the setting of the threshold in our investigations.

4. LEXICAL AND SYNTACTIC FEATURES

As stated previously, we restrict ourselves to lexical and syntactic features for this general investigation of the parameter space. These are reasonably easy to extract automatically for these texts and are different enough to determine if good parameter settings remain good when another type of feature is used. In further research, the choice of features is likely to increase.

4.1 Lexical Features

Sufficiently frequent tokens, that is those that were observed at least a certain amount of times (in this case 5) in some language reference corpus (in this case, the Eindhoven corpus [uit den Boogaart 1975] are used as features by themselves. For less frequent tokens, we determine a token pattern consisting of the sequence of character types, for instance, the token “Uefa-cup” is represented by the pattern #L#6+/CL-L, where the first L indicates low frequency, 6+ the size bracket, and the sequence CL-L a capital letter followed by one or more lower case letters, followed by a hyphen, and again one or more lower case letters. For lower case words, the final three letters of the word are included too, for instances, “waarmaken” leads to #L#6+/L/ken. These patterns were originally designed for English and Dutch and will probably have to be extended when other languages are handled.

In addition to the form of the token, we also use the potential syntactic usage of the token as a feature. We apply the first few modules of a morphosyntactic tagger (in this case, Wotan-Lite [van Halteren et al. 2001]) to the text, which determine which word-class tags could apply to each token. For known words, the tags are taken from a lexicon; for unknown words, they are estimated on the basis of the word patterns previously described. The three (if present) most likely tags are combined into a feature, for example, "niet" leads to #H#Adv(stell,onverv)-N(ev,neut) and "waarmaken" to #L#V(inf)-N(mv,neut)-V(verldw, onverv). Note that the most likely tags are determined on the basis of the token itself and that the context is not consulted. The modules of the tagger which do context-dependent disambiguation are not applied.

On top of the individual token and tag features, we use all possible bigrams and trigrams which can be built with them, for instances, the token combination "kon niet waarmaken" leads to features such as wcw= #H#kon#H#Adv(stell,onverv)-N(ev,neut)-#L#6+/L/ken. Since the number of features quickly grows too high for efficient processing, we filter the set of features by demanding that a feature occurs in a set minimum number of texts in the profile reference corpus (in this case two). A feature which is filtered out contributes instead to a rest category feature, for instance, the aforementaioned feature would contribute to wcw=<OTHER>. For the current corpus, this filtering leads to a feature set of about 100K features.

The lexical features currently also include features for utterance length. Each utterance leads to two such features, that is, the exact length (e.g., len=15) and the length bracket (e.g., len=10-19).

4.2 Syntactic Features

We used the Amazon parser to derive syntactic constituent analyses of each utterance [Coppen 2003]. We did not use the full rewrites but rather constituent N-grams. The N-grams used were:

(1) left-hand side label, examining constituent occurrence,
(2) left-hand side label plus one label from the right-hand side, examining dominance,
(3) left-hand side plus label two labels from the right-hand side, in their actual order, examining dominance and linear precedence.

For each label, two representations are used. The first is only the syntactic constituent label, the second is the constituent label plus the head word. This is done for each part of the N-grams independently, leading to 2, 4, and 8 features, respectively, for the three types of N-gram. Furthermore, each feature is used once by itself, once with an additional marking for the depth of the rewrite in the analysis tree, once with an additional marking for the length of the rewrite, and once with both these markings. This means another multiplication factor of four for a total of 8, 16, and 32 features, respectively. After filtering for a minimum number of observations, again at least an observation in two different texts, there are about 900K active syntactic features, nine times as many as for the lexical features.

Table I. Measurement Results at Optimal Settings for Each Feature Type and Optimization Criterion

	EER	$FAR_{FRR=0}$	ROC Surface	F
LEX	0.051	0.121	0.0159	0.855
SYN	0.090	0.181	0.0389	0.738
Combo(LEX,SYN)	**0.031**	**0.054**	0.0101	**0.894**
Combo(LEX,LEX)	**0.030**	0.075	**0.0077**	**0.898**

4.3 Combining Lexical and Syntactic Features

We are using separate profiles for lexical and syntactic features. However, we expect that a profile with access to both types of features should produce better results. For computational reasons, we did not actually calculate new combined profiles but just combined the classification scores from the two existing systems. Furthermore, to investigate whether it is the combination of lexical and syntactic features or merely the combination of two different models which is yielding a performance increase, we also used combinations of two systems both using lexical features. Finally, we did not experiment with second-level classifiers but merely added the scores from the two individual systems to investigate if combination indeed improves the quality of the classification.

5. RESULTS FOR THE VERIFICATION SCENARIO

The main scenario for this investigation is the verification one: we derive a model for a specific author and then test if this model can distinguish between texts by this author and texts by other authors. The relevant measures are equal error rate (EER), the false accept rate at the point where the false reject rate reaches zero, and the surface under the ROC curve. In addition, we examine the optimal F-measure.

5.1 Optimal Results for the Various Optimization Criteria

The best results for the various measures per feature type are shown in Table I. The general quality of the systems is good and the changes to the linguistic profiling technique have indeed been an improvement. If we compare the $FAR_{FRR=0}$ scores in van Halteren [2004] to the new ones, we see that the lexical features went from 14.9% to 12.1% (19% error reduction), the syntactic ones from 24.8% to 18.1% (27% error reduction), and the combination from 8.1% to 5.4% (33% error reduction).

In all cases, lexical features yield better results than syntactic ones. The best combination systems outperform the best individual systems, but there is no simple quality order between lexical-syntactic combination (clearly better on $FAR_{FRR=0}$) and lexical-lexical combination (clearly better on ROC surface). Generally, which settings bring the best performance differs with the choice of features. Moreover, the best settings are not merely determined by the choice of features, but also by other situational variables, such as the choice of evaluation criterion, as can be seen in Table II.

Optimizing for one criterion always leads to decreased performance on other criteria. The reason is that each optimization criterion needs a different focus

Table II. Measurement Results for All Optimization Criteria When Using Overall Optimal Settings for Each Individual Optimization Criterion

	Settings	EER	$FAR_{FRR=0}$	ROC surface	F
Best EER	C(L/P1.0/X8/P0.5, L/P1.2/P3.0/P-1.0) ABS	**0.030**	0.333	0.0162	0.886
Best $FAR_{FRR=0}$	C(L/P0.6/X8/P-0.5, S/P2.4/P-0.5/P0.5) REL	0.047	**0.054**	0.0164	0.842
Best ROC-surface	C(L/P0.6/X2/P0.5, L/P1.2/P2.5/P-0.5) ABS	0.042	0.403	**0.0077**	0.881
Best F	C(L/P1.0/X1/P0.5, L/P1.2/P2.5/P-0.5) ABS	0.046	0.458	0.0090	**0.898**

for the system. For a good EER, we need to separate the scores into two halves in such a way that there are as few texts as possible in the wrong half. How far a text model combination is into the wrong half does not really matter. For a good $FAR_{FRR=0}$, on the other hand, we need to get the worst scoring true text to score as high as possible (relatively) so that as few as possible false texts are accepted. For the ROC surface, it is harder to describe what is needed as the surface can be minimized in various ways. However, it seems likely that here we should aim to keep both the high and low regions free of false accepts/rejects. The F-value, finally, is mainly different because it belongs to the precision-recall viewpoint and rewards correct acceptance more than correct rejection. This also explains why the F-score looks worse than the other scores. After all, in our experiments, there are seven times as many texts that should be rejected than texts that should be accepted. The correct treatment of these is rewarded more from the verification viewpoint (FAR/FRR) than in the information retrieval viewpoint (precision/recall). Seeing the kind of applications we have in mind, the lower F-score does not temper our enthusiasm about the high other scores.

As for future actual application, our observations mean that it is not advisable to try to build a single system (model + settings + threshold) for all situations. Instead, one should decide what the best measure and optimization criterion is for each specific application and then try to train and tune a system for exactly that criterion (in our investigations, we have arbitrarily selected EER as the main criterion).

Note that, for all evaluation criteria, we presented the best result when choosing a single parameter setting and threshold for all runs. In a verification scenario, it might be more realistic if we assume that it is possible to vary these per author, or maybe even per model built. We investigated the effects of author-specific settings, although only for lexical features. The numbers of false rejects and false accepts per author (on a total of 9/63 positives/negatives) are shown in Table III.

As can be expected, the more settings can be varied per author or model, the better the performance. The problem, of course, is that we present the results for the best possible choices, and the question is whether we are able to find these choices for each author and/or model. We will not look into this question.

It seems that, with this technique, some authors appear to be easier to recognize than others, author 2 being the easiest. This is not so much visible in

Table III. Numbers of False Rejects and False Accepts per Author (on a total of 9/63 positives/negatives per author) for Parameter Setting Strategies Which are Increasingly More Adapted to the Specific Author

Author	Parameters Constant, Threshold Constant	Parameters Constant, Threshold per Author	Parameters per Author, Threshold per Author	Parameters per Author, Threshold per Model
1	0/1	0/1	0/0	0/0
2	0/9	0/2	0/0	0/0
3	0/1	0/0	0/0	0/0
4	1/2	1/1	0/4	0/0
5	0/6	0/0	0/0	0/0
6	1/1	0/3	0/1	0/0
7	1/3	0/7	0/4	0/1
8	0/7	0/3	0/0	0/0
Total	3/30	1/17	0/9	0/1
Total rates	0.042/0.060	0.014/0.034	0.000/0.018	0.000/0.002

the table as well in the number of parameter settings at which the 0/0 score in the second column from the right is reached: 1212 of the examined 7560 parameter settings result in this score. At least for this author, this gives us good hope that parameter settings might be derived automatically by training on the known author texts. At the other extreme, we find authors 4 and 7. For author 7, even the best settings per model are plagued by the fact that there is a topic where the text by a different author has a higher score than the text by author 7, leading to an unavoidable false accept.

5.2 Potential for Training

Suppose we are faced with texts by a new author and we have to create a new model for this author. Can we then automatically find the parameters and a threshold for this model? One option is to reuse the settings which have proven their worth with previous authors. However, as we have already shown, varying the parameter settings per author leads to much better results than using single settings. Settings for other authors do not seem all that promising then. Since building a model for a new author implies that we have at least some known texts for this author, it seems to be more sensible to try to derive good settings from these texts. We have attempted to investigate whether the excellent optimal scores can be translated into good (or at least acceptable) trained scores. For this exercise, we restricted ourselves to lexical features and used only equal error rate as an evaluation criterion.

For each run of the 9-fold cross-validation described in Section 3.2, we trained a model on the 8 other topics, just as in the previous experiments. However, we also used this training data as if it was all the data we had and performed an 8-fold cross-validation experiment with it. For these 8 topics, we then searched for the optimal parameter settings and threshold for each author. We then used these settings and threshold with the full 8-topic model to classify the 9th topic text.

The results of this classification are shown in the second column of Table IV. The overall score is a false reject rate of 9.7% and a false accept rate of 5.2%. In

Table IV. Numbers of False Rejects and False Accepts per Author (on a total of 9/63 positives/Negatives per author) When Parameter Settings are Trained Rather Than Optimal

Author	Parameters Trained, Threshold Trained	Parameters Trained, Threshold Optimal
1	1/2	0/1
2	0/0	0/0
3	1/1	0/0
4	4/5	0/4
5	0/2	0/0
6	0/9	0/0
7	0/7	0/2
8	1/0	0/1
Total	7/26	0/8
Total rates	0.097/0.052	0.000/0.016

comparison to the optimal single setting scores from Table III (4.2% and 6.0%), this seems a very acceptable result. However, since we are using settings and threshold per author, we should compare to the fourth column of Table III (0.0% and 1.8%), which means our trained setting quality is quite far away from the optimal setting ceiling. Still, if we examine where the differences stem from, we see that it is especially the threshold which needs improvement. If we use an optimal threshold for each classification, we find the results in the third column of Table IV, which show an overall score of FRR 0.0% and FAR 1.6%. As for the results per author, again we find that author 2 can be recognized very well and authors 4 and 7 are the most difficult. For the others, there is some variation with regard to the earlier recognition ease, but we should note that the data is, of course, very limited.

So we find that it is especially the threshold which is hard to determine correctly. This could have been expected in that the test run uses another model (trained on 8 topics) than the training runs (trained on 7 topics), leading to unavoidable differences in verification scores. Would it not be better then to use the 7-topic training runs directly? We examined this as well. For each 7 topics, we trained a model and determined all parameter settings per author for which an 8th topic text by this author was at the best possible rank (in practice, always rank 1). For each such setting, we selected a threshold value which distinguished between the author text and the other texts. To have a safety margin, we set the threshold at 9/10th of the gap between the score for the author text and the first next other-author text. We then used these settings and the corresponding thresholds, still with the same 7-topic model, to classify the 9th topic text. In this way, we derived a large number of classifications for each text (8 models times however many settings yielded the best rank for the tuning texts). Beforehand, we intuitively set an overall acceptance threshold at 50% of the classifications, that is, if a text would be accepted in at least half of the large number of classifications, we would accept it as being written by the claimed author. The overall verification results are shown in Table V.

The overall score for our preselected threshold (second column) is an FRR of 6.9% and an FAR of 4.6%, slightly better than the 9.7% and 5.2% of the standard

Table V. Numbers of False Rejects and False Accepts per Author (on a total of 9/63 positives/negatives per author) When Using a Multiple Comparison Strategy Instead of Training for Single Parameter Settings

Author	Final Threshold at 50%	Final Threshold at 54%	Final Threshold Optimal per Author
1	0/4	0/3	0/3
2	0/2	0/0	0/0
3	0/1	0/0	0/0
4	2/5	2/3	0/13
5	0/3	0/1	0/1
6	0/1	0/2	0/1
7	2/4	2/2	2/2
8	1/3	1/1	0/5
Total	5/23	5/11	2/25
Total rates	0.069/0.046	0.069/0.022	0.028/0.050

training scenario. However, we are once again faced with a threshold selection, as can be seen from the third and fourth columns. This, added to the fact that the overall procedure is so much more complicated per new text to be verified, leads us to prefer the previous once-off setting determination.

5.3 An Observation on Combination

As described previously, when training for a new author, we use n-fold cross-training to determine what we expect to be good parameters. For each of the runs, we measure how well each setting is performing and then choose the overall best setting. This is already computationally expensive for just the lexical features. If we want to use a combination of any kind, even a combination with only two models as building blocks, it means testing millions of models. It would be very welcome if we could decrease the computation time by measuring only combinations of the most promising individual components. However, we would then assume that good combinations can only be built with good building blocks. To test this assumption, we examined the relations between building block quality and combination quality.

Figure 1 shows the average EER of the two building blocks in lexical-lexical combination as a function of the EER for the combination. We see that there is a very rough correlation between the two. However, the bottom-left corner of the plot, where the better EERs are found, seems to deviate from the general trend; especially, the best combination does not seem to be formed with the best building blocks.

This is confirmed when we examine the bottom-left corner in more detail. Figures 2(a) and 2(b) show the EER for the best-scoring and the worst-scoring component in each combination pair as a function of the combination EER. The very best combination (EER = 3.0%) is built with components with individual EERs of 12.9% and 28.7%. Other top combinations also appear to be formed with the best components showing an EER from the 10–15% range, more than twice that of the best individual model (EER = 5.1%). Even more interesting is that the worst components in the best combinations have EERs of 25–30% and that quite good combinations can be made with components having EERs of

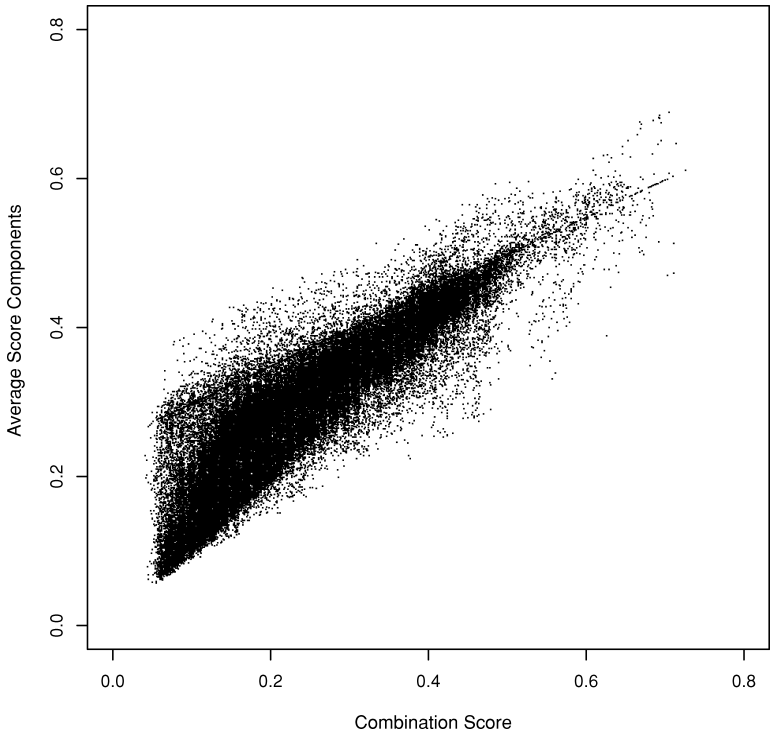

Fig. 1. Average EER of the two building blocks in a lexical-lexical combination as a function of the EER for the combination.

over 60%. It is not quality but complementarity which makes good combination partners. One might even say that the worst component in some cases acts only as a kind of correction factor to the best component, and hence does not need to be good by itself at all.

We see the same behavior for the lexical-syntactic combinations. We will not present plots, as they have much the same shape, but only give some numbers. The best lexical component (EER = 5.1%) and the best syntactic component (EER = 9.0%) lead to a combination with an EER of 6.1%, about twice that of the best lexical-syntactic combination (EER = 3.1%). Furthermore, this combination is actually performing worse than the lexical component by itself, so that examining only these components would make us decide not to use combination at all. It we do look further, we find that the best combination (EER 3.1%) is built from a lexical component with EER 12.7% and a syntactic component with EER 22.2%, both more than twice the EER of the corresponding best component and ranked around individual positions 1000 (lexical) and 1540 (syntactic). So if we want to find the best combination, we will have to examine a very large proportion of all possible combinations after all.

6. RESULTS FOR THE RECOGNITION SCENARIO

We have focused on the authorship verification task, since it is the harder problem, given that the potential group of authors is unknown. However, as

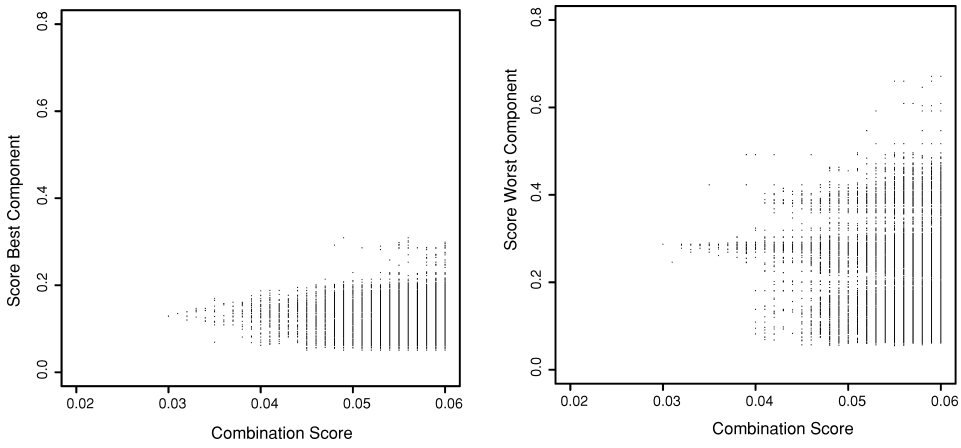

Fig. 2. EER of the best-scoring (2(a)) and worst-scoring (2(b)) building blocks in a lexical-lexical combination as a function of the EER for the combination.

Table VI. Best Achieved Authorship Recognition Quality for Various Methods

	2-way Errors /504	2-way Percent Correct	8-way Errors /72	8-way Percent Correct
50 function words, PCA		± 50%		
followed by LDA		± 60%		
LDA with cross-sample entropy weighting		± 80%		
all tokens, WPDV modeling		97.8%		
LP2004 (lex)	6	98.8%	5	93%
LP2004 (syn)	14	97.2%	10	86%
LP2004 (lex+syn)	3	99.4%	2	97%
LP2005 (lex)	4	99.2%	3	96%
LP2005 (syn)	12	97.6%	8	89%
LP2005 (lex+syn)	2	99.6%	2	97%
LP2005 (lex+lex)	0	100%	0	100%

mentioned in Section 2, most earlier work with this data has focused on the authorship recognition problem, to be exact, on selecting the correct author out of two potential authors. This is possible with linguistic profiling, too, by calculating the author scores for each author from the set for a given text, and then selecting the author with the highest score. We list the optimal results, together with the previously published results in Table VI, both for the 2-way and for the 8-way selection problem.

Again, the newer Linguistic Profiling system outperforms the previous one. We also see that the optimal settings are different than those for the verification scenario, in other words, they have changed with the selection of a new optimization criterion. However, this is not really surprising since here we compare scores across models rather than within models.

7. CONCLUSION

Linguistic profiling has certainly shown its worth for authorship recognition and verification. At the best single overall settings found so far, a profiling system using a combination of scores for different settings is able to select the correct author for all texts in the test corpus. It is also able to perform the verification task in such a way that it rejects no texts that should be accepted, while accepting only 5.4% of the texts that should be rejected, or alternatively yielding an equal error rate of 3%. Allowing settings to vary per author leads to even better verification quality. However, it should be noted that all these scores are for optimal settings and that, although optimal parameters turn out to be quite trainable, the threshold so far still resists automatic determination.

The next step in the investigation of linguistic profiling, then, is a continued search for a training procedure for that remaining elusive setting, the acceptance threshold. Another avenue of future research is the inclusion of even more types of features, not only those which we had envisioned ourselves, but also those showing good results in other systems (e.g., Koppel et al. [2003] and Kešelj et al. [2003]). However, it would be useful to define an even harder verification task, as the current system already scores very high and further improvements might be hard to measure. With the current corpus, the task might be made harder by limiting the size of the test texts.

Other corpora might also serve to provide more obstinate data, although it must be said that the current test corpus was designed specifically for this purpose. Use of further corpora will also help finetune the training procedures as they may show the similarities and/or differences in behavior between datasets. Finally, with the right types of corpora, the worth of the technique for actual application scenarios could be investigated.

So there are several possible routes to further improvement. Still, the current quality of the system is such that the system can be applied as is. For authorship recognition and verification, it (or rather its previous incarnation) performed well in both our own experiments and in the forementioned contest. For language verification, van Halteren and Oostdijk [2004] showed good results. And other text classification tasks could now possibly also be investigated, such as language or language variety recognition, genre recognition, or document classification for IR purposes.

REFERENCES

BAAYEN, R. H., VAN HALTEREN, H., NEIJT, A., AND TWEEDIE, F. 2002. An experiment in authorship attribution. In *Proceedings of the International Conference in Textual Data Statistical Analysis (JADT)*. 69–75.

BROEDERS, T. 2001. Forensic speech and audio analysis, forensic linguistics 1998-2001—A review. In *Proceedings of the 13th Interpol Forensic Science Symposium* Lyon, France.

CHASKI, C. 2001. Empirical evaluations of language-based author identification techniques. *Forensic Linguistics 8*, 1, 1–65.

COPPEN, P. A. 2003. Rejuvenating the amazon parser. *Computational Linguistics in the Netherlands (CLIN'03),* Antwerp, Belgium (Dec).

VAN HALTEREN, H. 2004. Linguistic profiling for author recognition and verification. In *Proceedings of the Association for Computational Linguisticss. (ACL04).* 200–207.

VAN HALTEREN, H., ZAVREL, J., AND DAELEMANS, W. 2001. Improving accuracy in word class tagging through the combination of machine learning systems. *Computat. Linguistics 27*, 2, 199–230.

VAN HALTEREN, H., AND OOSTDIJK, N. 2004. Linguistic profiling of texts for the purpose of language verification. In *Proceedings of the International Conference on Computation Linguistics (COLING'04)*. 966–972.

VAN HALTEREN, H., HAVERKORT, M., BAAYEN, R. H., NEIJT, A., AND TWEEDIE, F. 2005. New machine learning methods demonstrate the existence of a human stylome. *J. Quantitat. Linguistics 12*, 1, 65–78.

HOLMES, D. 1998. Authorship attribution. *Literary Linguistic Comput. 13*, 3, 111–117.

KEŠELJ, V., PENG, F., CERCONE, N., AND THOMAS, C. 2003. N-gram-based author profiles for authorship attribution. In *Proceedings of the Pacific Conference Association for Computational Linguistics (PACLING'03)*. Halifax, Nova Scotia, Canada (Aug.).

KOPPEL, M., AND SCHLER, J. 2003. Exploiting stylistic idiosyncrasies for authorship attribution. In *Proceedings of Workshop on Computational Approaches to Style Analysis and Synthesis, (IJCAI'03).* Acapulco, Mexico.

KOPPEL, M., AKIVA, N., AND DAGAN, I. 2003. A corpus-independent feature set for style based text categorization. In *Proceedings of Workshop on Computational Approaches to Style Analysis and Synthesis (IJCAI'03).* Acapulco, Mexico.

MOSTELLER, F. AND WALLACE, D. L. 1984. *Applied Bayesian and Classical Inference in the Case of the Federalist Papers*, 2nd ed. Springer Verlag, Berlin, Germany.

UIT DEN BOOGAART, P. C. 1975. *Woordfrequenties in geschreven en gesproken Nederlands*. Oosthoek, Scheltema & Holkema, Utrecht.

Received November 2005; accepted August 2006 by Kishore Papineni