Authorship Verification via k-Nearest Neighbor Estimation

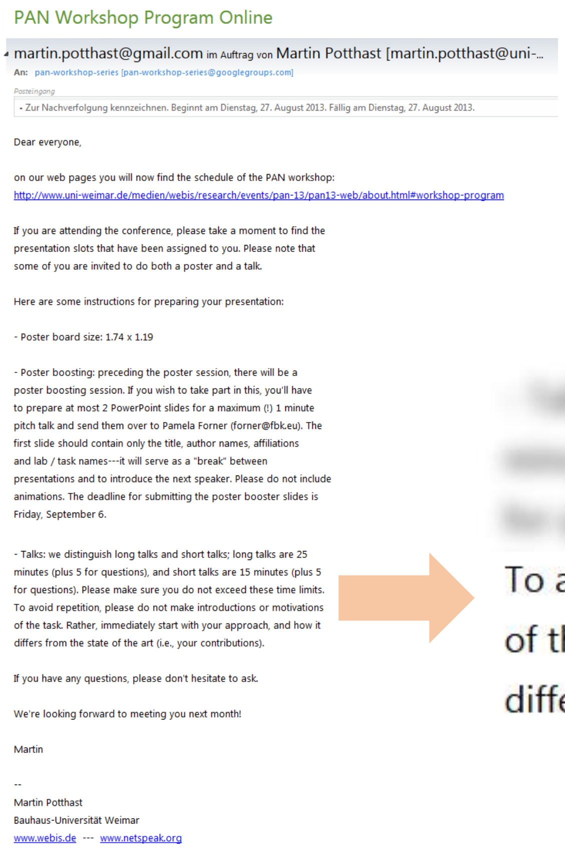
Oren Halvani, Martin Steinebach, Ralf Zimmermann

Fraunhofer Institute for Secure Information Technology (SIT), Darmstadt, Germany
Department of Computer Science Technische Universität Darmstadt, Germany

OUTLINE

- Verification schemes
- Features & Feature-Categories
- Our approach
- Evaluation
- Benefits / challenges / future work

MOTIVATION



PAN Workshop Program Online

martin.potthast@gmail.com im Auftrag von Martin Potthast [martin.potthast@uni-...

An: pan-workshop-series [pan-workshop-series@googlegroups.com]

Posteingang

- Zur Nachverfolgung kennzeichnen. Beginnt am Dienstag, 27. August 2013. Fällig am Dienstag, 27. August 2013.

Dear everyone,

on our web pages you will now find the schedule of the PAN workshop:
http://www.uni-weimar.de/medien/webis/research/events/pan-13/pan13-web/about.html#workshop-program

If you are attending the conference, please take a moment to find the presentation slots that have been assigned to you. Please note that some of you are invited to do both a poster and a talk.

Here are some instructions for preparing your presentation:

- Poster board size: 1.74 x 1.19

- Poster boosting: preceding the poster session, there will be a poster boosting session. If you wish to take part in this, you'll have to prepare at most 2 PowerPoint slides for a maximum (!) 1 minute pitch talk and send them over to Pamela Forner (forner@fbk.eu). The first slide should contain only the title, author names, affiliations and lab / task names---it will serve as a "break" between presentations and to introduce the next speaker. Please do not include animations. The deadline for submitting the poster booster slides is Friday, September 6.

- Talks: we distinguish long talks and short talks; long talks are 25 minutes (plus 5 for questions), and short talks are 15 minutes (plus 5 for questions). Please make sure you do not exceed these time limits. To avoid repetition, please do not make introductions or motivations of the task. Rather, immediately start with your approach, and how it differs from the state of the art (i.e., your contributions).

If you have any questions, please don't hesitate to ask.

We're looking forward to meeting you next month!

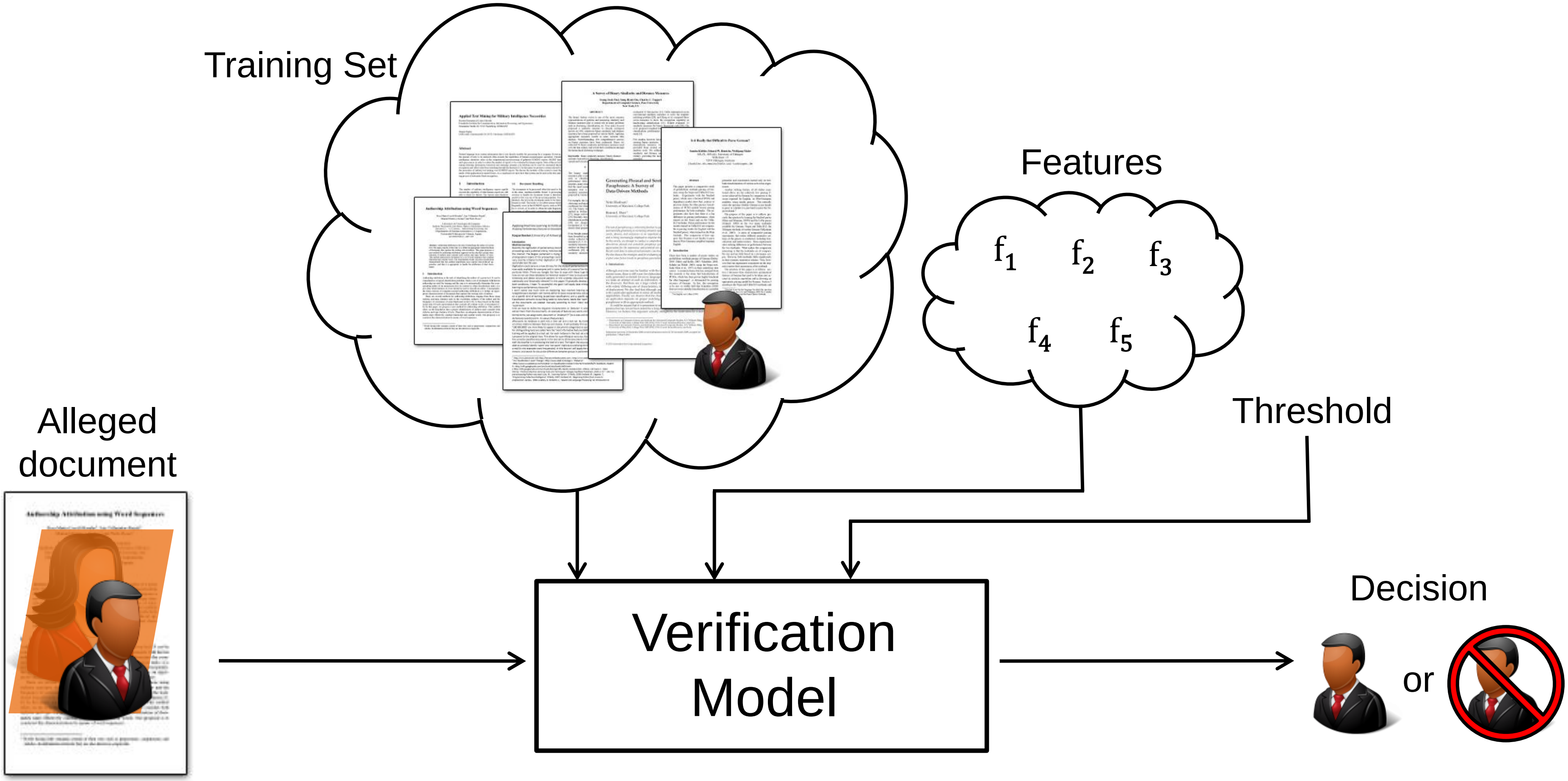
Martin

--
Martin Potthast
Bauhaus-Universität Weimar
www.webis.de --- www.netspeak.org

- Talks: we distinguish long talks and short talks; long talks are 25 minutes (plus 5 for questions), and short talks are 15 minutes (plus 5 for questions). Please make sure you do not exceed these time limits. To avoid repetition, please do not make introductions or motivations of the task. Rather, immediately start with your approach, and how it differs from the state of the art (i.e., your contributions).

☺ So, let's start immediately…

VERIFICATION SCHEME (CLASSICAL VERSION…)



Training Set

Features

f_1 f_2 f_3 f_4 f_5

Threshold

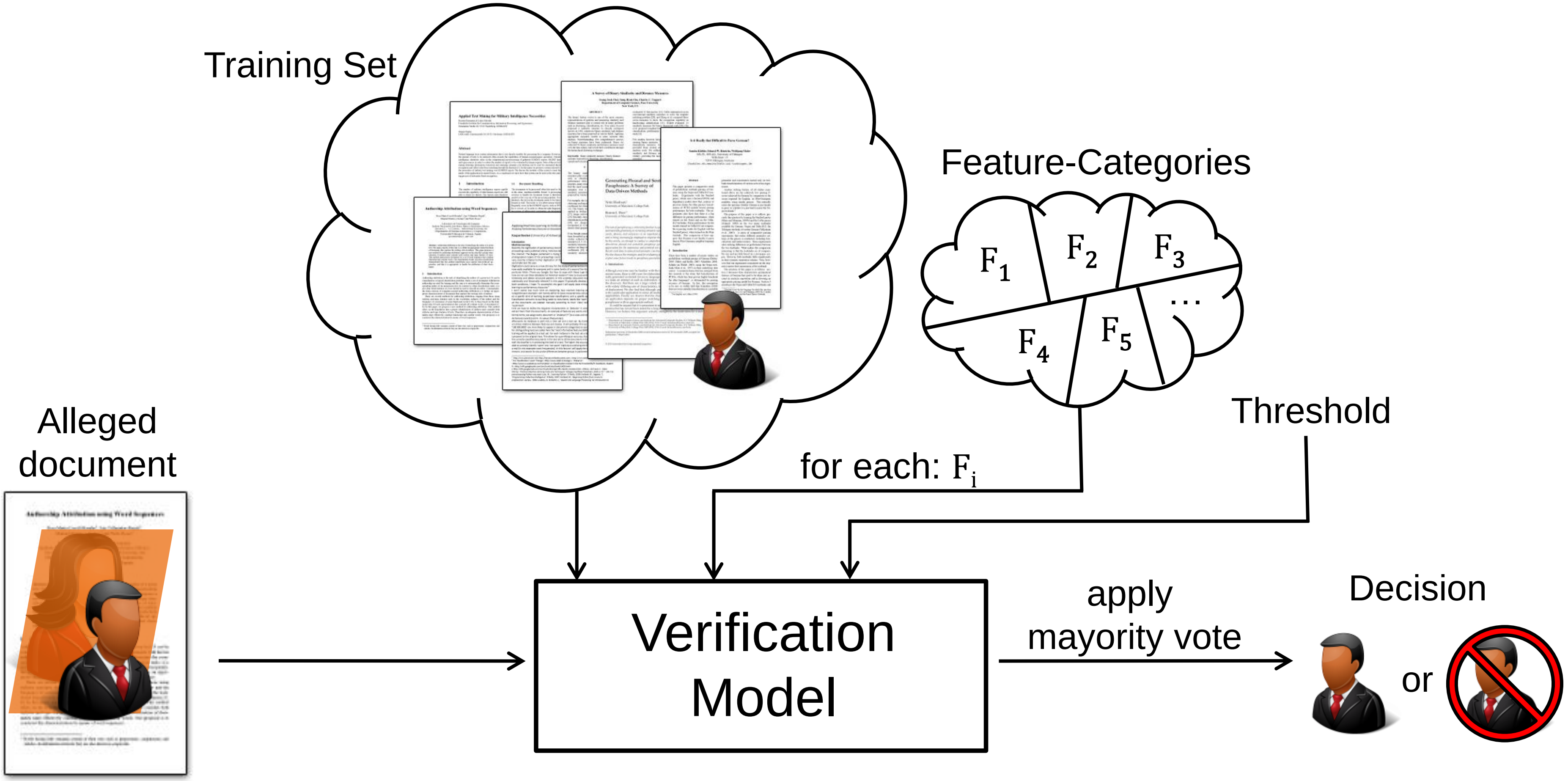
Alleged document

Verification Model

Decision

or

VERIFICATION SCHEME (OUR VERSION…)



Training Set

Feature-Categories

F_1 F_2 F_3 F_4 F_5 …

Alleged document

for each: F_i

Threshold

Verification Model

apply mayority vote

Decision

or

FEATURES

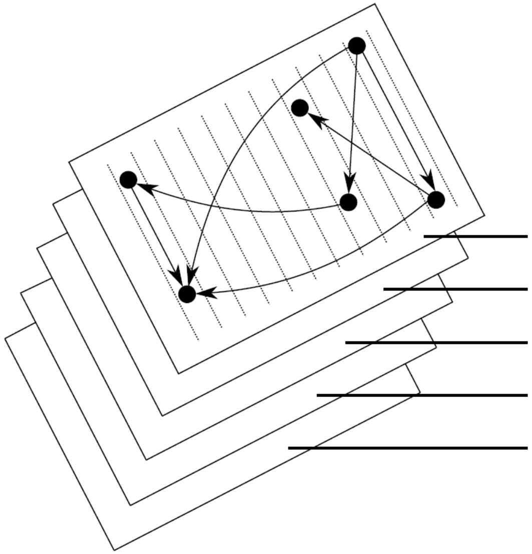
- Features are the core of any AV system!

FEATURES

- Features are the core of any AV system!
- Usually classified into so-called linguistic layers (e.g. survey of Stamatatos)

FEATURES

- Features are the core of any AV system!
- Usually classified into so-called linguistic layers (e.g. survey of Stamatatos)



Semantic layer

Syntactic layer

Lexical layer

Character layer

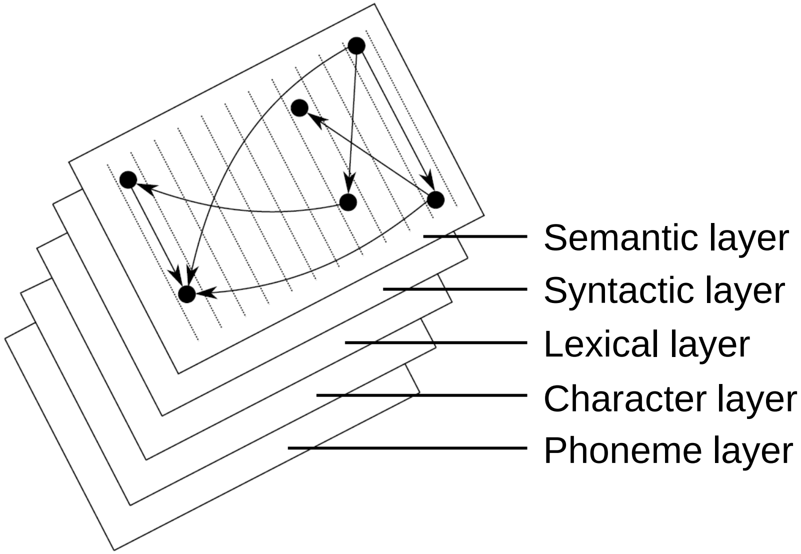
Phoneme layer

There are even more, e.g. Layout layer

FEATURES

- Features are the core of any AV system!

- Usually classified into so-called linguistic layers (e.g. survey of Stamatatos)



There are even more, e.g. Layout layer

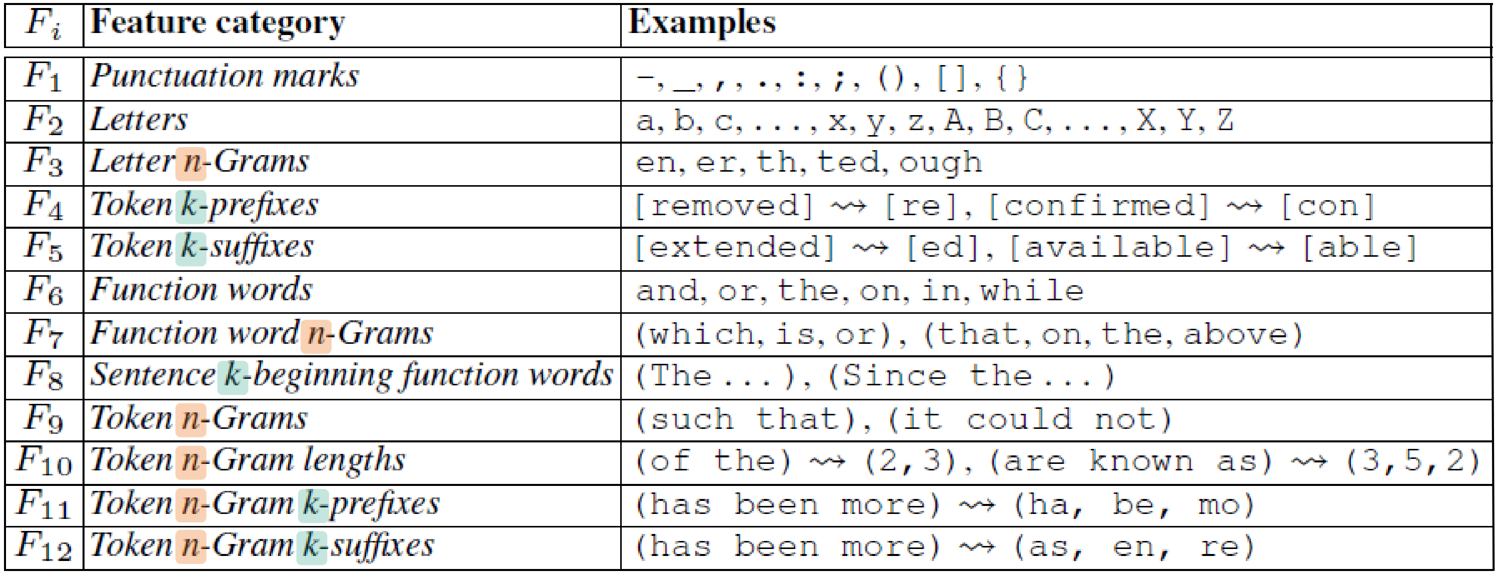
- Instead of "layers" we prefer to use the term "Feature-Categories"…

FEATURE CATEGORIES

- We understand a "Feature-Category" as a concept of features, belonging to (at least) one linguistic layer…

FEATURE CATEGORIES

- We understand a "Feature-Category" as a concept of features, belonging to (at least) one linguistic layer…



F_i	**Feature category**	**Examples**
F_1	*Punctuation marks*	-, _, ,, ., :, ;, (), [], {}
F_2	*Letters*	a, b, c, . . . , x, y, z, A, B, C, . . . , X, Y, Z
F_3	*Letter n-Grams*	en, er, th, ted, ough
F_4	*Token k-prefixes*	[removed] ⇝ [re], [confirmed] ⇝ [con]
F_5	*Token k-suffixes*	[extended] ⇝ [ed], [available] ⇝ [able]
F_6	*Function words*	and, or, the, on, in, while
F_7	*Function word n-Grams*	(which, is, or), (that, on, the, above)
F_8	*Sentence k-beginning function words*	(The . . .), (Since the . . .)
F_9	*Token n-Grams*	(such that), (it could not)
F_{10}	*Token n-Gram lengths*	(of the) ⇝ (2,3), (are known as) ⇝ (3,5,2)
F_{11}	*Token n-Gram k-prefixes*	(has been more) ⇝ (ha, be, mo)
F_{12}	*Token n-Gram k-suffixes*	(has been more) ⇝ (as, en, re)

FEATURE-CATEGORIES (PARAMETERS)

- **Note:** Majority of these Feature-Categories can be parameterized…

FEATURE-CATEGORIES (PARAMETERS)

- **Note:** Majority of these Feature-Categories can be parameterized…
  - *n-Gram sizes*
  - *k-prefix / suffixes*
  - *Amount of dictionary based features*
  - *etc.*

FEATURE-CATEGORIES (PARAMETERS)

- **Note:** Majority of these Feature-Categories can be parameterized…
  - *n-Gram sizes*
  - *k-prefix / suffixes*
  - *Amount of dictionary based features*
  - *etc.*

- **Moreover:** Frequencies of extracted features are also kept variable (e.g. „*use the 120 most frequent letter-bigrams*“)

FEATURE-CATEGORIES (PARAMETERS)

- **Note:** Majority of these Feature-Categories can be parameterized…

  - *n-Gram sizes*
  - *k-prefix / suffixes*
  - *Amount of dictionary based features*
  - *etc.*

- **Moreover:** Frequencies of extracted features are also kept variable (e.g. „*use the 120 most frequent letter-bigrams*“)

- **Consequence:** Practically unlimited parameter space!

FEATURE-CATEGORIES (PARAMETERS)

- **Note:** Majority of these Feature-Categories can be parameterized…
  - *n-Gram sizes*
  - *k-prefix / suffixes*
  - *Amount of dictionary based features*
  - *etc.*
- **Moreover:** Frequencies of extracted features are also kept variable (e.g. „*use the 120 most frequent letter-bigrams*“)
- **Consequence:** Practically unlimited parameter space!
- **(Unsatisfactory) solution:** random examination…

OUR APPROACH

- The procedure of our AV system can be divided into three steps:

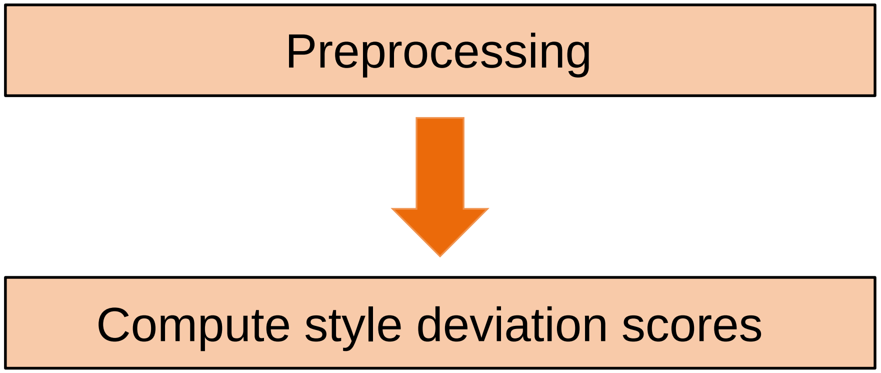
OUR APPROACH

- The procedure of our AV system can be divided into three steps:

Preprocessing

OUR APPROACH

- The procedure of our AV system can be divided into three steps:

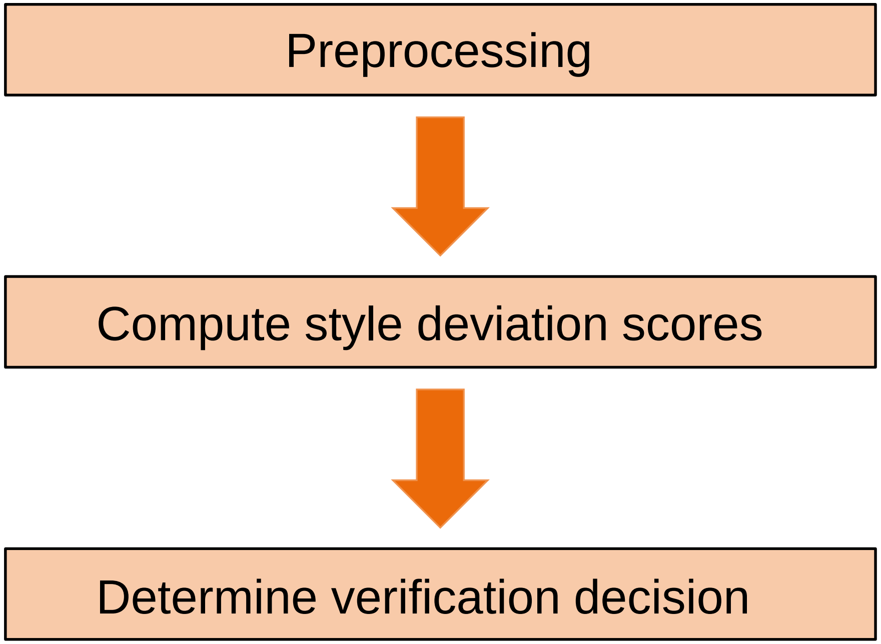


Preprocessing

Compute style deviation scores

OUR APPROACH

- The procedure of our AV system can be divided into three steps:


Preprocessing

Compute style deviation scores

Determine verification decision

OUR APPROACH: PREPROCESSING

- Applying preprocessing in terms of **normalization** and **noise reduction**

OUR APPROACH: PREPROCESSING

- Applying preprocessing in terms of **normalization** and **noise reduction**

Essential to treat all documents uniquely!

→ e.g. substituting diacritics, successive blanks, etc.

OUR APPROACH: PREPROCESSING

- Applying preprocessing in terms of **normalization** and **noise reduction**

Essential to treat all documents uniquely!

→ e.g. substituting diacritics, successive blanks, etc.

Important to increase quality of extracted features!

→ e.g. removing citations, markup-tags, formulas, non-words, etc.

OUR APPROACH:
COMPUTE STYLE DEVIATION SCORES

- Our approach is based on a k-Nearest Neighbours (k-NN) classifier

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Our approach is based on a k-$\mathrm{Nearest\ Neighbours}$ (k-NN) classifier
- Hence, we need to construct feature-vectors from Y and X_1, X_2, ..., X_m
  → for each chosen Feature-Category…

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Our approach is based on a k-Nearest Neighbours (k-NN) classifier

- Hence, we need to construct feature-vectors from $\mathbf{Y}$ and $\mathbf{X}_1$, $\mathbf{X}_2$, ..., $\mathbf{X}_m$
→ for each chosen Feature-Category…

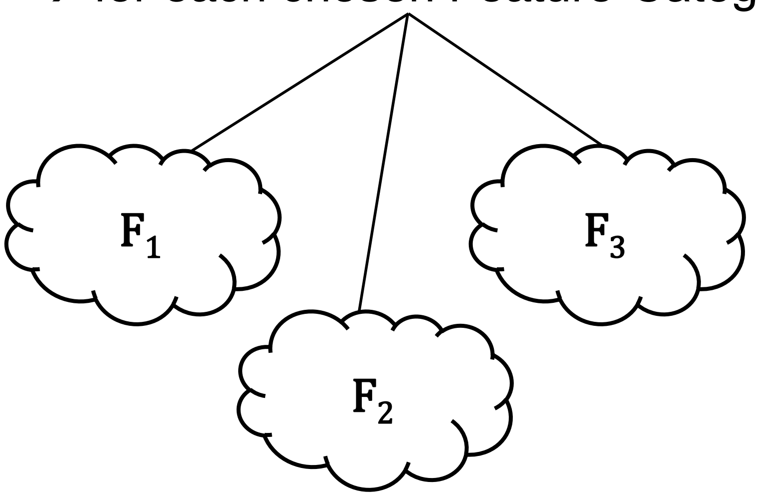
Alleged document

All documents from the training set

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Our approach is based on a k-Nearest Neighbours (k-NN) classifier

- Hence, we need to construct feature-vectors from Y and X_1, X_2, ..., X_m
  → for each chosen Feature-Category…



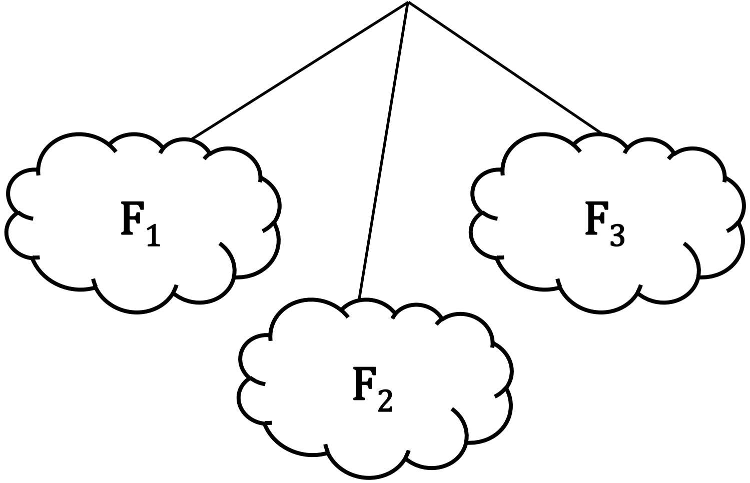
Alleged document

All documents from the training set

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Our approach is based on a k-Nearest Neighbours (k-NN) classifier

- Hence, we need to construct feature-vectors from $\mathbf{Y}$ and $\mathbf{X}_1$, $\mathbf{X}_2$, ..., $\mathbf{X}_m$
  → for each chosen Feature-Category…



Alleged document

All documents from the training set

- **Important:** Majority-voting needs an uneven number of individual decisions
  → hence, number of $\mathbf{F}_i$ is always odd

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- We calculate pairwise style deviation scores (SDS) between Y and X_1, X_2, …, X_m for each chosen $\mathbf{F}_\mathrm{i}$

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- We calculate pairwise style deviation scores (SDS) between Y and X_1, X_2, ..., X_m for each chosen $\mathbf{F}_\mathrm{i}$

- A SDS is a number between $[0 - \infty)$, which is calculated through a distance function, e.g. Euclidean distance:

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- We calculate pairwise style deviation scores (SDS) between $\mathbf{Y}$ and $\mathbf{X}_1$, $\mathbf{X}_2$, ..., $\mathbf{X}_m$ for each chosen $\mathbf{F}_\mathrm{i}$

- A SDS is a number between $[0 - \infty)$, which is calculated through a distance function, e.g. Euclidean distance:

$$dist_{Euclid}(X,Y) = \sqrt{\sum_{i=1}^{n} \left(x_i - y_i\right)^2}$$

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- We calculate pairwise style deviation scores (SDS) between Y and X_1, X_2, ..., X_m for each chosen $\mathbf{F}_i$

- A SDS is a number between [0 - ∞), which is calculated through a distance function, e.g. Euclidean distance:

$$dist_{Euclid}(X,Y) = \sqrt{\sum_{i=1}^{n}\left(x_i - y_i\right)^2}$$

- The closer a SDS is to zero, the more similar X_i is to Y

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- We calculate pairwise style deviation scores (SDS) between Y and X_1, X_2, ..., X_m for each chosen $\mathbf{F}_i$

- A SDS is a number between [0 - ∞), which is calculated through a distance function, e.g. Euclidean distance:

$$dist_{Euclid}(X,Y) = \sqrt{\sum_{i=1}^{n}\left(x_i - y_i\right)^2}$$

- The closer a SDS is to zero, the more similar X_i is to Y

- Once all SDS's are calculated we've got to store them…

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Resulting SDS's are stored together with the corresponding feature vectors into a sorted list (ascending order, according to the scores)

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Resulting SDS's are stored together with the corresponding feature vectors into a sorted list (ascending order, according to the scores)

$$\boxed{Outer\_Distances = \left((\mathrm{SDS}_1, \mathbf{X}_1), (\mathrm{SDS}_2, \mathbf{X}_2), \ldots, (\mathrm{SDS}_m, \mathbf{X}_m) \right)}$$

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Resulting SDS's are stored together with the corresponding feature vectors into a sorted list (ascending order, according to the scores)

$$Outer\_Distances = \left((\mathrm{SDS}_1, \mathrm{X}_1), (\mathrm{SDS}_2, \mathrm{X}_2), \ldots, (\mathrm{SDS}_m, \mathrm{X}_m) \right)$$

- Next, we extract the first tuple and calculate again SDS's but now between X_1 and $\mathrm{X}_2, \mathrm{X}_3, \ldots, \mathrm{X}_m$

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Resulting SDS’s are stored together with the corresponding feature vectors into a sorted list (ascending order, according to the scores)

$$\boxed{\mathit{Outer\_Distances} = \left((\mathrm{SDS}_1, \mathbf{X}_1), (\mathrm{SDS}_2, \mathbf{X}_2), \ldots, (\mathrm{SDS}_m, \mathbf{X}_m) \right)}$$

- Next, we extract the first tuple and calculate again SDS’s but now between $\mathbf{X}_1$ and $\mathbf{X}_2, \mathbf{X}_3, \ldots, \mathbf{X}_m$

- Now we store only the SDS‘s into another ordered list:

OUR APPROACH: COMPUTE STYLE DEVIATION SCORES

- Resulting SDS's are stored together with the corresponding feature vectors into a sorted list (ascending order, according to the scores)

$$Outer\_Distances = \left((\mathrm{SDS}_1, \mathrm{X}_1), (\mathrm{SDS}_2, \mathrm{X}_2), \ldots, (\mathrm{SDS}_m, \mathrm{X}_m) \right)$$

- Next, we extract the first tuple and calculate again SDS's but now between X_1 and X_2, X_3, …, X_m

- Now we store only the SDS's into another ordered list:

$$Inner\_Distances = \left(\mathrm{SDS}_2, \mathrm{SDS}_3, \ldots, \mathrm{SDS}_m \right)$$

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the k-SDS‘s within $Inner\_Distances$:

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the k-SDS‘s within *Inner_Distances*:

$$avg\_SDS = \frac{\mathrm{SDS}_2 + \mathrm{SDS}_3 + \cdots + \mathrm{SDS}_\mathrm{k}}{\mathrm{k}}$$

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the *k*-*SDS*'s within *Inner_Distances*:

$$avg\_SDS = \frac{\mathrm{SDS}_2 + \mathrm{SDS}_3 + \cdots + \mathrm{SDS}_\mathrm{k}}{\mathrm{k}}$$

k-NN of X_1

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the *k*-*SDS*'s within *Inner_Distances*:

$$avg\_SDS = \frac{\mathrm{SDS}_2 + \mathrm{SDS}_3 + \cdots + \mathrm{SDS}_\mathrm{k}}{\mathrm{k}}$$

k-NN of X_1

- Now we can define an acceptance criterion

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the *k*-*SDS*'s within *Inner_Distances*:

$$avg\_SDS = \frac{\mathrm{SDS}_2 + \mathrm{SDS}_3 + \cdots + \mathrm{SDS}_\mathrm{k}}{\mathrm{k}}$$

k-NN of X_1

- Now we can define an acceptance criterion

- Accept the alleged authorship if…

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the *k-SDS*‘s within *Inner_Distances*:

$$avg\_SDS = \frac{\mathrm{SDS}_2 + \mathrm{SDS}_3 + \cdots + \mathrm{SDS}_k}{\mathrm{k}}$$

k-NN of X_1

- Now we can define an acceptance criterion

- Accept the alleged authorship if…

$$\frac{\mathrm{SDS}_1}{\mathrm{avg\_SDS}} \leq \mathrm{Threshold}$$

CASED

Fraunhofer SIT

OUR APPROACH: DETERMINE VERIFICATION DECISION

- To obtain a decision regarding a chosen feature category we first calculate the average of the *k-SDS*‘s within *Inner_Distances*:

$$avg\_SDS = \frac{\mathrm{SDS}_2 + \mathrm{SDS}_3 + \cdots + \mathrm{SDS}_k}{\mathrm{k}}$$

k-NN of X_1

- Now we can define an acceptance criterion

- Accept the alleged authorship if…

$$\frac{\mathrm{SDS}_1}{\mathrm{avg\_SDS}} \leq \mathrm{Threshold}$$

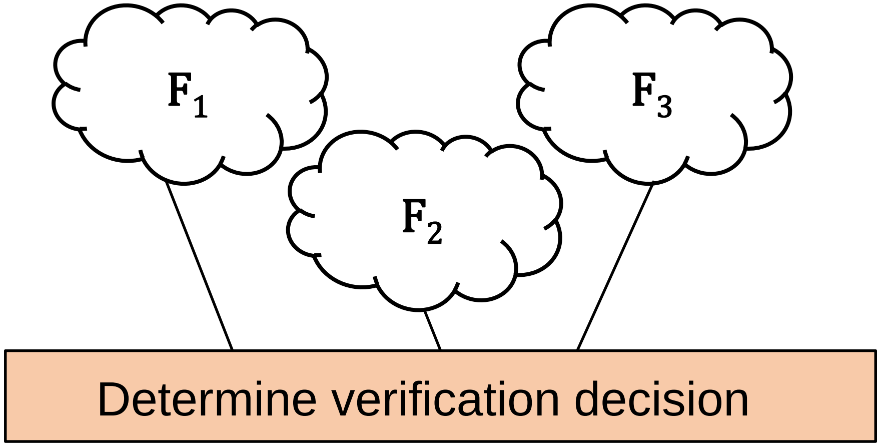
In most of the cases: ***1*** *performs very well…*

OUR APPROACH:
DETERMINE VERIFICATION DECISION

- Overall decision regarding all Feature-Categories would then be:

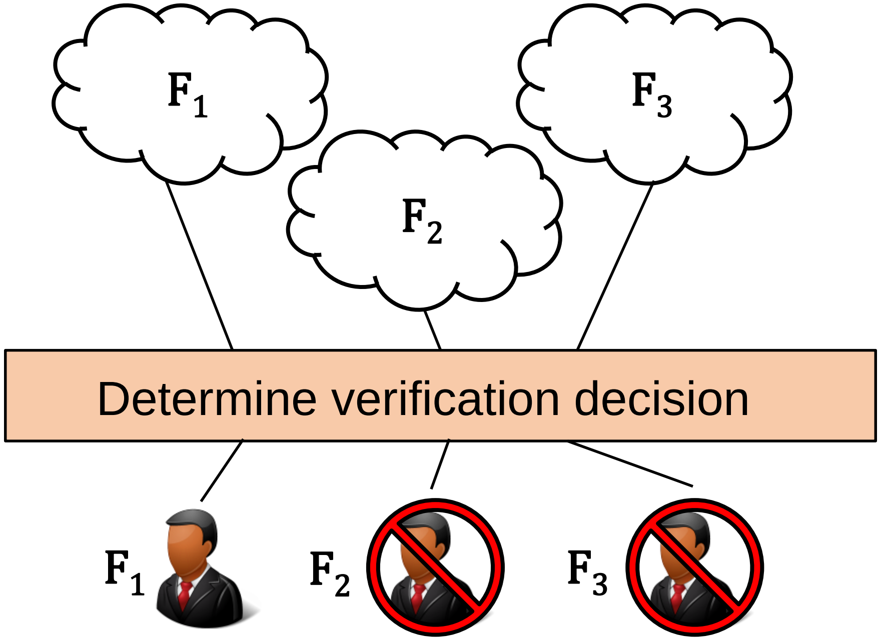
OUR APPROACH: DETERMINE VERIFICATION DECISION

- Overall decision regarding all Feature-Categories would then be:



OUR APPROACH: DETERMINE VERIFICATION DECISION

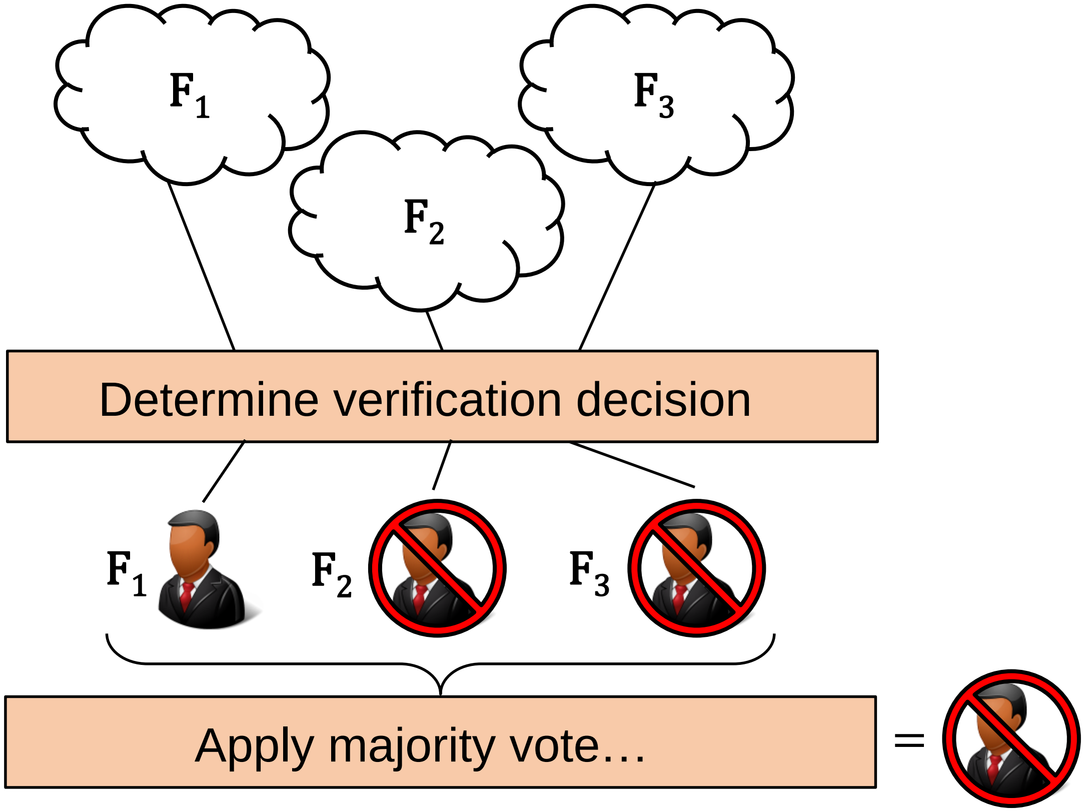
- Overall decision regarding all Feature-Categories would then be:



Determine verification decision

OUR APPROACH: DETERMINE VERIFICATION DECISION

- Overall decision regarding all Feature-Categories would then be:



Determine verification decision

Apply majority vote…

EVALUATION: USED MEASURES

Simple accuracy:

$$\varnothing = \frac{\varnothing_{\mathcal{C}_{GR}} + \varnothing_{\mathcal{C}_{EN}} + \ldots}{|\mathcal{C}_{GR} \cup \mathcal{C}_{EN} \cup \ldots|} \text{ , with } \varnothing_{\mathcal{C}_i} = \frac{\text{Number of correct answers per dataset } \mathcal{C}_i}{\text{Total number of documents per dataset } \mathcal{C}_i}$$

EVALUATION: USED MEASURES

Simple accuracy:

$$\varnothing = \frac{\varnothing_{\mathcal{C}_{GR}} + \varnothing_{\mathcal{C}_{EN}} + \ldots}{|\mathcal{C}_{GR} \cup \mathcal{C}_{EN} \cup \ldots|} \text{ , with } \varnothing_{\mathcal{C}_i} = \frac{\text{Number of correct answers per dataset } \mathcal{C}_i}{\text{Total number of documents per dataset } \mathcal{C}_i}$$

Weighted accuracy:

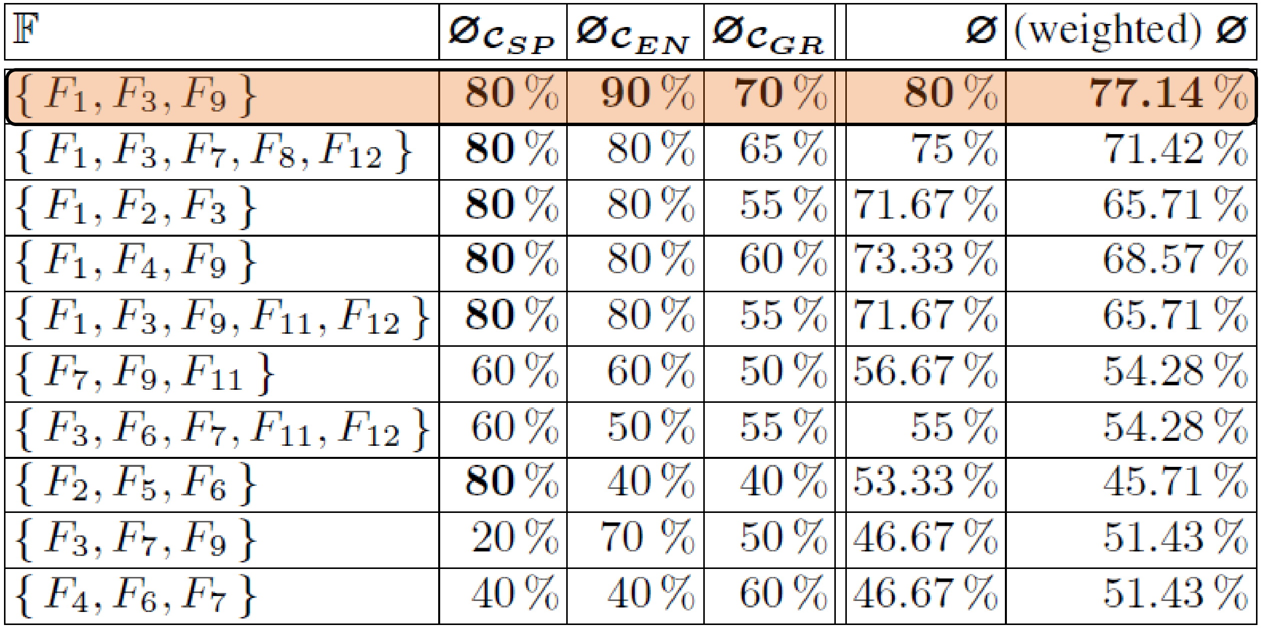
$$(\text{weighted})\varnothing = \frac{|\mathcal{C}_{GR}| \cdot \varnothing_{\mathcal{C}_{GR}} + |\mathcal{C}_{EN}| \cdot \varnothing_{\mathcal{C}_{EN}} + \ldots}{|\mathcal{C}_{GR} \cup \mathcal{C}_{EN} \cup \ldots|}$$

EVALUATION: TRAIN SET (PAN ONLY)

- Evaluation results according to "PAN13-AI-Training Corpus"

EVALUATION: TRAIN SET (PAN ONLY)

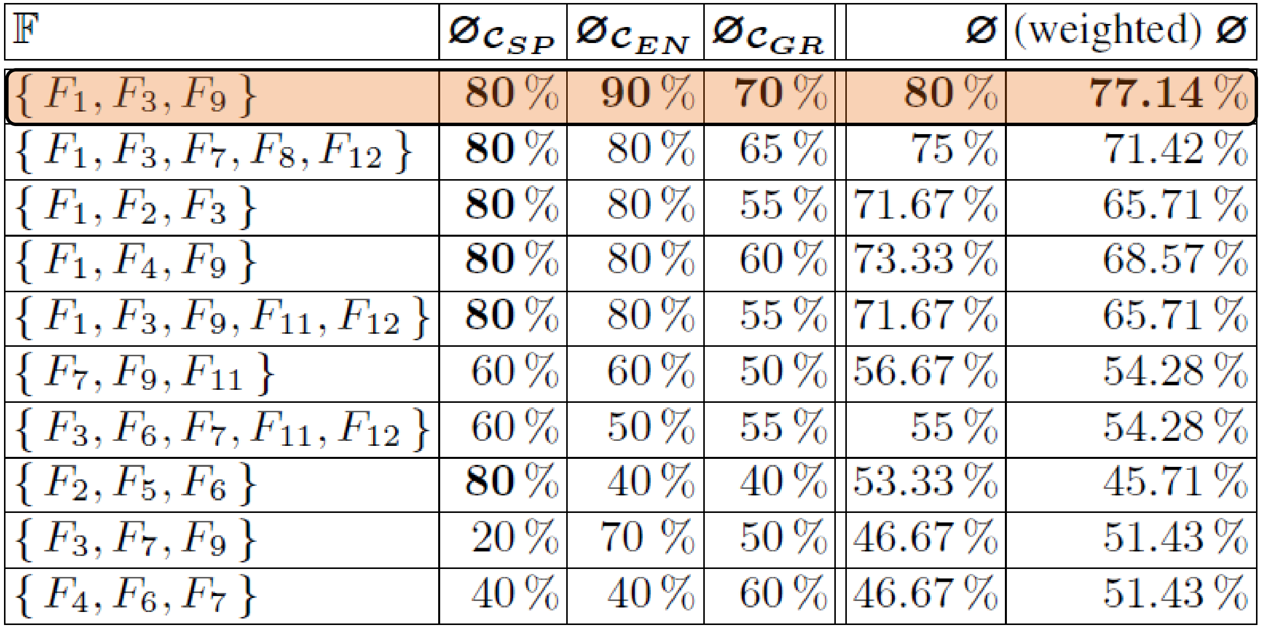
- Evaluation results according to "PAN13-AI-Training Corpus"



$\mathbb{F}$	$\varnothing c_{SP}$	$\varnothing c_{EN}$	$\varnothing c_{GR}$	$\varnothing$	(weighted) $\varnothing$
$\{ F_1, F_3, F_9 \}$	**80 %**	**90 %**	**70 %**	**80 %**	**77.14 %**
$\{ F_1, F_3, F_7, F_8, F_{12} \}$	**80 %**	80 %	65 %	75 %	71.42 %
$\{ F_1, F_2, F_3 \}$	**80 %**	80 %	55 %	71.67 %	65.71 %
$\{ F_1, F_4, F_9 \}$	**80 %**	80 %	60 %	73.33 %	68.57 %
$\{ F_1, F_3, F_9, F_{11}, F_{12} \}$	**80 %**	80 %	55 %	71.67 %	65.71 %
$\{ F_7, F_9, F_{11} \}$	60 %	60 %	50 %	56.67 %	54.28 %
$\{ F_3, F_6, F_7, F_{11}, F_{12} \}$	60 %	50 %	55 %	55 %	54.28 %
$\{ F_2, F_5, F_6 \}$	**80 %**	40 %	40 %	53.33 %	45.71 %
$\{ F_3, F_7, F_9 \}$	20 %	70 %	50 %	46.67 %	51.43 %
$\{ F_4, F_6, F_7 \}$	40 %	40 %	60 %	46.67 %	51.43 %

EVALUATION: TRAIN SET (PAN ONLY)

- Evaluation results according to "PAN13-AI-Training Corpus"



$\mathbb{F}$	$\varnothing c_{SP}$	$\varnothing c_{EN}$	$\varnothing c_{GR}$	$\varnothing$	(weighted) $\varnothing$
$\{F_1, F_3, F_9\}$	**80 %**	**90 %**	**70 %**	**80 %**	**77.14 %**
$\{F_1, F_3, F_7, F_8, F_{12}\}$	**80 %**	80 %	65 %	75 %	71.42 %
$\{F_1, F_2, F_3\}$	**80 %**	80 %	55 %	71.67 %	65.71 %
$\{F_1, F_4, F_9\}$	**80 %**	80 %	60 %	73.33 %	68.57 %
$\{F_1, F_3, F_9, F_{11}, F_{12}\}$	**80 %**	80 %	55 %	71.67 %	65.71 %
$\{F_7, F_9, F_{11}\}$	60 %	60 %	50 %	56.67 %	54.28 %
$\{F_3, F_6, F_7, F_{11}, F_{12}\}$	60 %	50 %	55 %	55 %	54.28 %
$\{F_2, F_5, F_6\}$	**80 %**	40 %	40 %	53.33 %	45.71 %
$\{F_3, F_7, F_9\}$	20 %	70 %	50 %	46.67 %	51.43 %
$\{F_4, F_6, F_7\}$	40 %	40 %	60 %	46.67 %	51.43 %

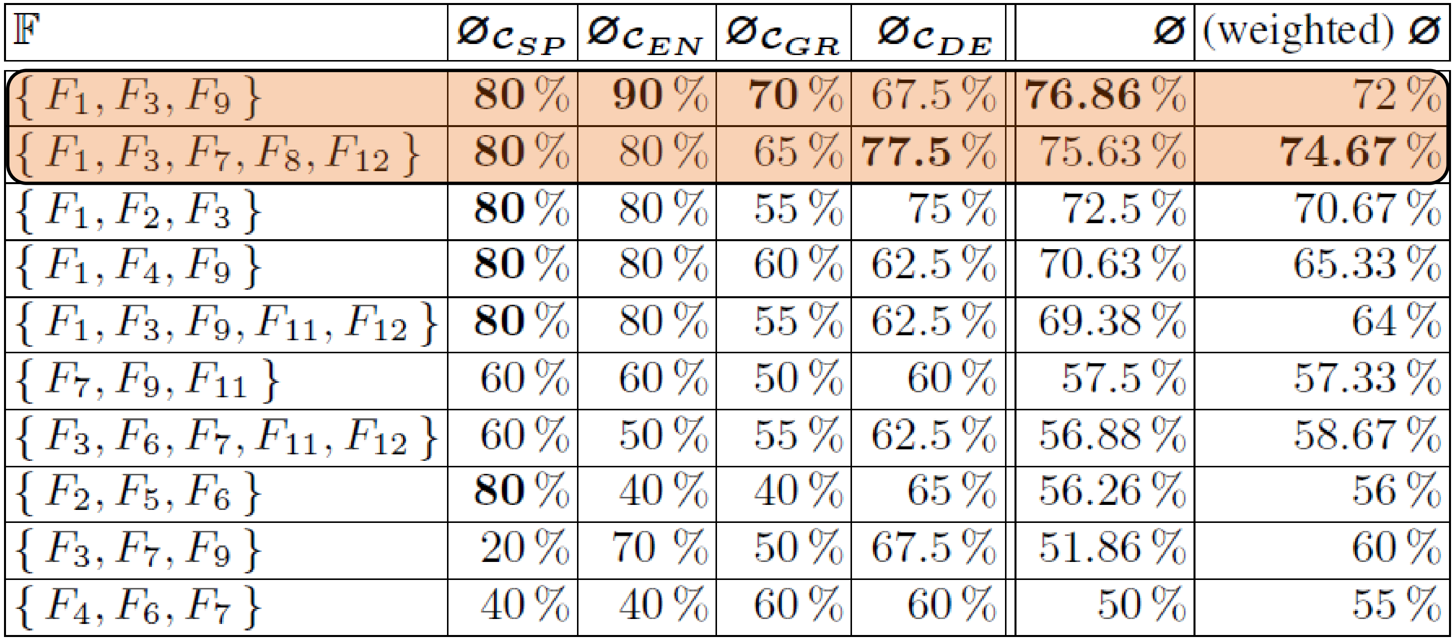
- **Note:** the first one is the **best** $\mathbf{F}_\mathrm{i}$ - combination out of **2^{12}** = **4096**

EVALUATION: TRAIN SET (PAN + GERMAN CORPUS)

- Evaluation results according to "PAN13-AI-Training Corpus" in addition to a self-compiled german corpus (40 problem-cases)

EVALUATION: TRAIN SET (PAN + GERMAN CORPUS)

- Evaluation results according to "PAN13-AI-Training Corpus" in addition to a self-compiled german corpus (40 problem-cases)



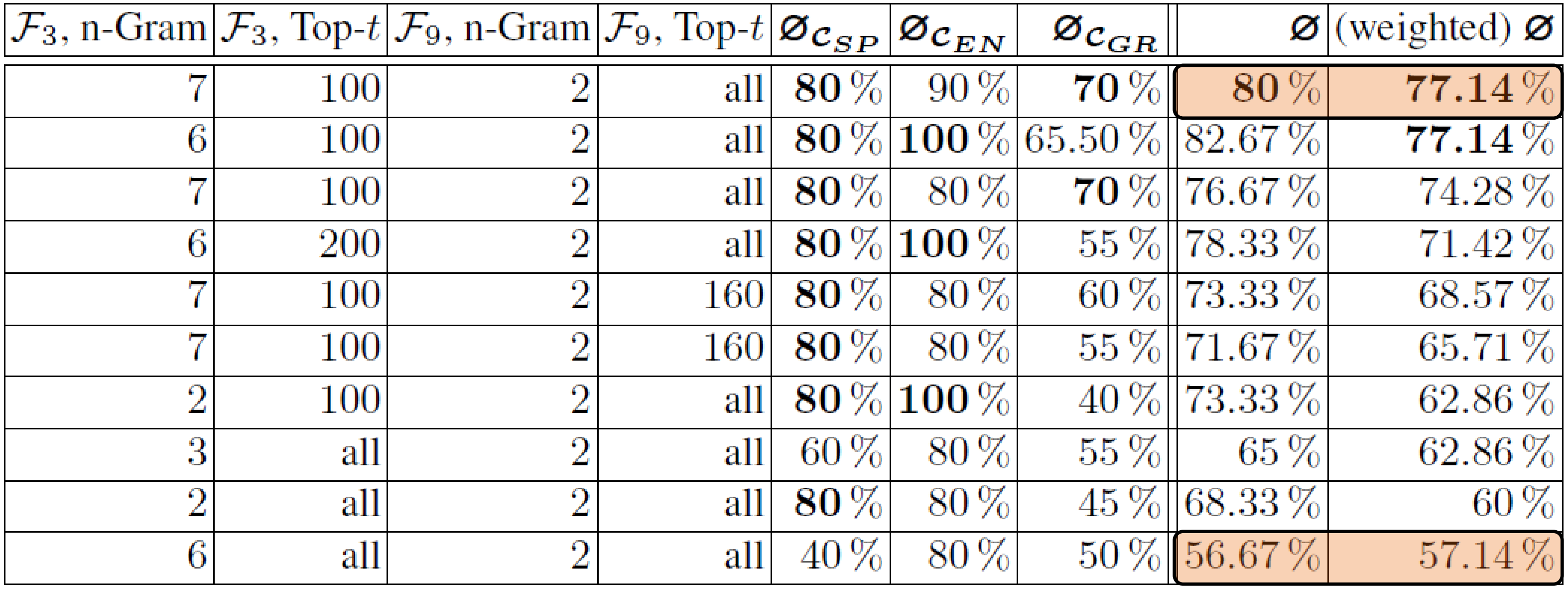
$\mathbb{F}$	$\varnothing c_{SP}$	$\varnothing c_{EN}$	$\varnothing c_{GR}$	$\varnothing c_{DE}$	$\varnothing$	(weighted) $\varnothing$
$\{F_1, F_3, F_9\}$	**80 %**	**90 %**	**70 %**	67.5 %	**76.86 %**	72 %
$\{F_1, F_3, F_7, F_8, F_{12}\}$	**80 %**	80 %	65 %	**77.5 %**	75.63 %	**74.67 %**
$\{F_1, F_2, F_3\}$	**80 %**	80 %	55 %	75 %	72.5 %	70.67 %
$\{F_1, F_4, F_9\}$	**80 %**	80 %	60 %	62.5 %	70.63 %	65.33 %
$\{F_1, F_3, F_9, F_{11}, F_{12}\}$	**80 %**	80 %	55 %	62.5 %	69.38 %	64 %
$\{F_7, F_9, F_{11}\}$	60 %	60 %	50 %	60 %	57.5 %	57.33 %
$\{F_3, F_6, F_7, F_{11}, F_{12}\}$	60 %	50 %	55 %	62.5 %	56.88 %	58.67 %
$\{F_2, F_5, F_6\}$	**80 %**	40 %	40 %	65 %	56.26 %	56 %
$\{F_3, F_7, F_9\}$	20 %	70 %	50 %	67.5 %	51.86 %	60 %
$\{F_4, F_6, F_7\}$	40 %	40 %	60 %	60 %	50 %	55 %

EVALUATION: TRAIN SET (PAN → INFLUENCE OF PARAMETERS)

- Evaluation results according to "PAN13-AI-Training Corpus" with the best combination $\{F_1, F_3, F_9\}$ nd various parameter-settings

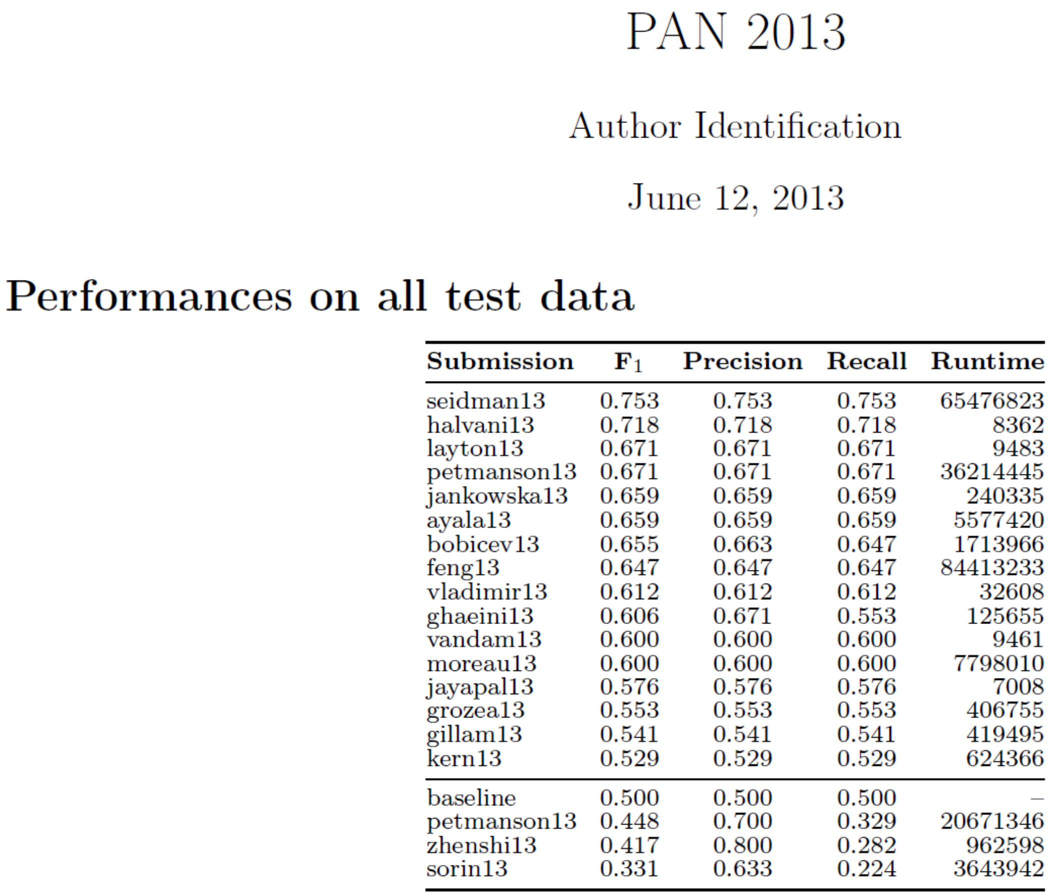
EVALUATION: TRAIN SET (PAN → INFLUENCE OF PARAMETERS)

- Evaluation results according to "PAN13-AI-Training Corpus" with the best combination $\{ F_1, F_3, F_9 \}$nd various parameter-settings



$\mathcal{F}_3$, n-Gram	$\mathcal{F}_3$, Top-t	$\mathcal{F}_9$, n-Gram	$\mathcal{F}_9$, Top-t	$\varnothing c_{SP}$	$\varnothing c_{EN}$	$\varnothing c_{GR}$	$\varnothing$	(weighted) $\varnothing$
7	100	2	all	**80** %	90 %	**70** %	**80** %	**77.14** %
6	100	2	all	**80** %	**100** %	65.50 %	82.67 %	**77.14** %
7	100	2	all	**80** %	80 %	**70** %	76.67 %	74.28 %
6	200	2	all	**80** %	**100** %	55 %	78.33 %	71.42 %
7	100	2	160	**80** %	80 %	60 %	73.33 %	68.57 %
7	100	2	160	**80** %	80 %	55 %	71.67 %	65.71 %
2	100	2	all	**80** %	**100** %	40 %	73.33 %	62.86 %
3	all	2	all	60 %	80 %	55 %	65 %	62.86 %
2	all	2	all	**80** %	80 %	45 %	68.33 %	60 %
6	all	2	all	40 %	80 %	50 %	56.67 %	57.14 %

EVALUATION: TEST SET



PAN 2013

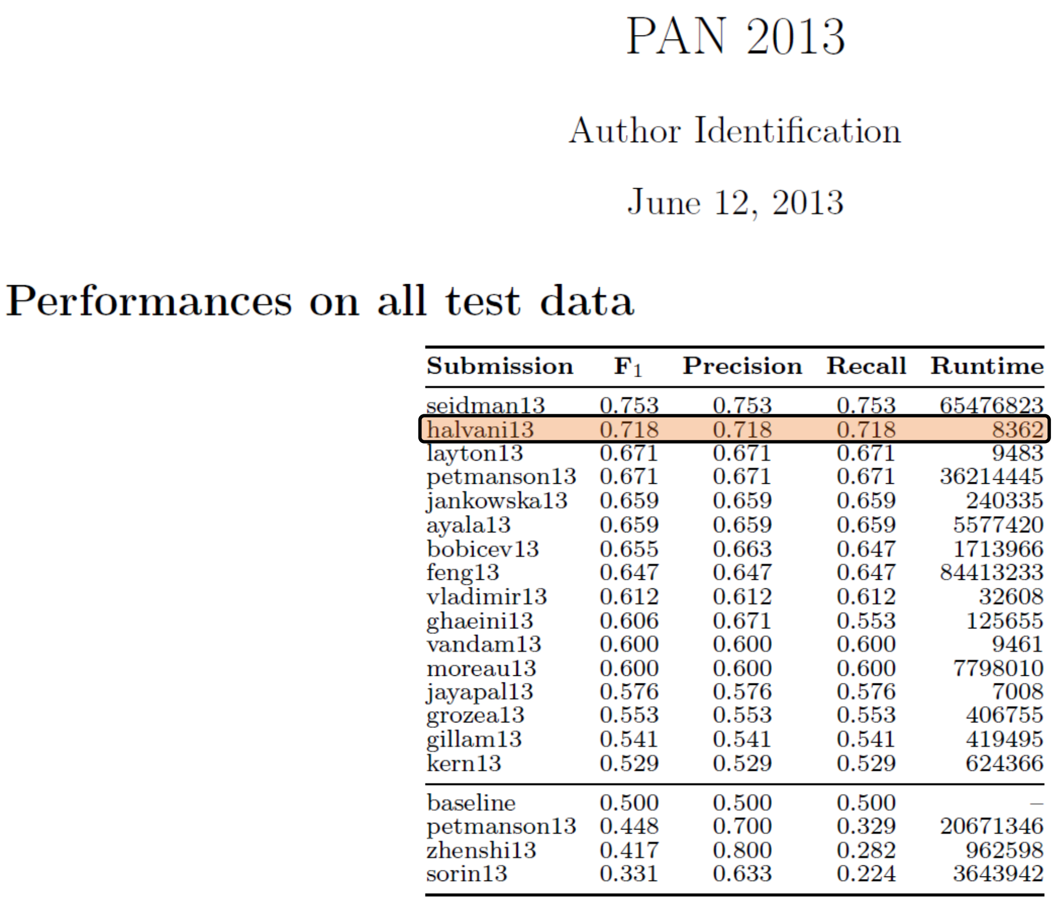
Author Identification

June 12, 2013

Performances on all test data

Submission	F_1	Precision	Recall	Runtime
seidman13	0.753	0.753	0.753	65476823
halvani13	0.718	0.718	0.718	8362
layton13	0.671	0.671	0.671	9483
petmanson13	0.671	0.671	0.671	36214445
jankowska13	0.659	0.659	0.659	240335
ayala13	0.659	0.659	0.659	5577420
bobicev13	0.655	0.663	0.647	1713966
feng13	0.647	0.647	0.647	84413233
vladimir13	0.612	0.612	0.612	32608
ghaeini13	0.606	0.671	0.553	125655
vandam13	0.600	0.600	0.600	9461
moreau13	0.600	0.600	0.600	7798010
jayapal13	0.576	0.576	0.576	7008
grozea13	0.553	0.553	0.553	406755
gillam13	0.541	0.541	0.541	419495
kern13	0.529	0.529	0.529	624366
baseline	0.500	0.500	0.500	–
petmanson13	0.448	0.700	0.329	20671346
zhenshi13	0.417	0.800	0.282	962598
sorin13	0.331	0.633	0.224	3643942

EVALUATION: TEST SET



PAN 2013

Author Identification

June 12, 2013

Performances on all test data

Submission	F$_1$	Precision	Recall	Runtime
seidman13	0.753	0.753	0.753	65476823
halvani13	0.718	0.718	0.718	8362
layton13	0.671	0.671	0.671	9483
petmanson13	0.671	0.671	0.671	36214445
jankowska13	0.659	0.659	0.659	240335
ayala13	0.659	0.659	0.659	5577420
bobicev13	0.655	0.663	0.647	1713966
feng13	0.647	0.647	0.647	84413233
vladimir13	0.612	0.612	0.612	32608
ghaeini13	0.606	0.671	0.553	125655
vandam13	0.600	0.600	0.600	9461
moreau13	0.600	0.600	0.600	7798010
jayapal13	0.576	0.576	0.576	7008
grozea13	0.553	0.553	0.553	406755
gillam13	0.541	0.541	0.541	419495
kern13	0.529	0.529	0.529	624366
baseline	0.500	0.500	0.500	–
petmanson13	0.448	0.700	0.329	20671346
zhenshi13	0.417	0.800	0.282	962598
sorin13	0.331	0.633	0.224	3643942

If runtime would count too… ☺

BENEFITS

- Our approach has several benefits, as for instance:

BENEFITS

- Our approach has several benefits, as for instance:

- **Language-independent**, but not cross-lingual, e.g.:
  Y is written in another language than X_1, X_2, ..., X_m

BENEFITS

- Our approach has several benefits, as for instance:

- **Language-independent**, but not cross-lingual, e.g.:
  Y is written in another language than X_1, X_2, ..., X_m

- **Very fast**, there's no need for time-consuming NLP-operations

BENEFITS

- Our approach has several benefits, as for instance:

- **Language-independent**, but not cross-lingual, e.g.:
  Y is written in another language than X_1, X_2, ..., X_m

- **Very fast**, there's no need for time-consuming NLP-operations

- **Scalable approach**, almost anything can be replaced, expanded or combined…

BENEFITS

- Our approach has several benefits, as for instance:
- **Language-independent**, but not cross-lingual, e.g.:
  Y is written in another language than X_1, X_2, ..., X_m
- **Very fast**, there's no need for time-consuming NLP-operations
- **Scalable approach**, almost anything can be replaced, expanded or combined…

Threshold, distance function(s), Feature-Categories (and their parameters),…

CHALLENGES / FUTURE WORK

- **Biggest challenge:**
  Inscrutability of the methods parameter-space ☹
  → Number of parameter-settings of the feature categories is near infinite

CHALLENGES / FUTURE WORK

- **Biggest challenge:**
  Inscrutability of the methods parameter-space ☹
  → Number of parameter-settings of the feature categories is near infinite

- **Possible solution:**
  Integrate evolutionary algorithms into the AV-system to find optimal parameter settings → bad run-time performance ☹

CHALLENGES / FUTURE WORK

- **Biggest challenge:**
  Inscrutability of the methods parameter-space ☹
  → Number of parameter-settings of the feature categories is near infinite

- **Possible solution:**
  Integrate evolutionary algorithms into the AV-system to find optimal parameter settings → bad run-time performance ☹

- **Another challenge:**
  Does the topic of the test (or training documents) has a **strong** influence on the classification result? → Still an open question…

CHALLENGES / FUTURE WORK

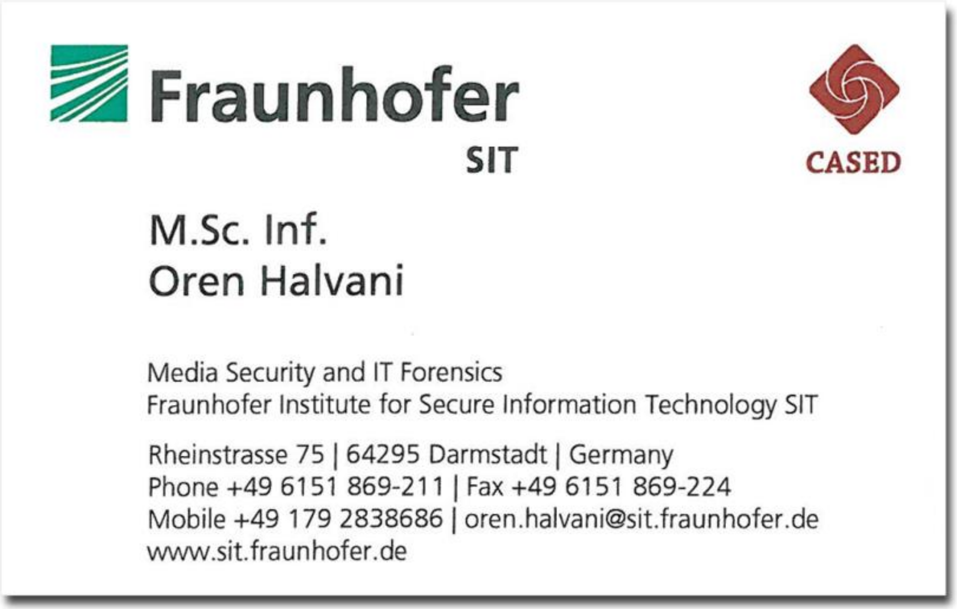
- **Biggest challenge:**
  Inscrutability of the methods parameter-space ☹
  → Number of parameter-settings of the feature categories is near infinite

- **Possible solution:**
  Integrate evolutionary algorithms into the AV-system to find optimal parameter settings → bad run-time performance ☹

- **Another challenge:**
  Does the topic of the test (or training documents) has a **strong** influence on the classification result? → Still an open question…

- **Possible Solution:**
  One of our students is currently writing his thesis to answer this question

Thank you very much for your attention!



Fraunhofer
SIT

CASED

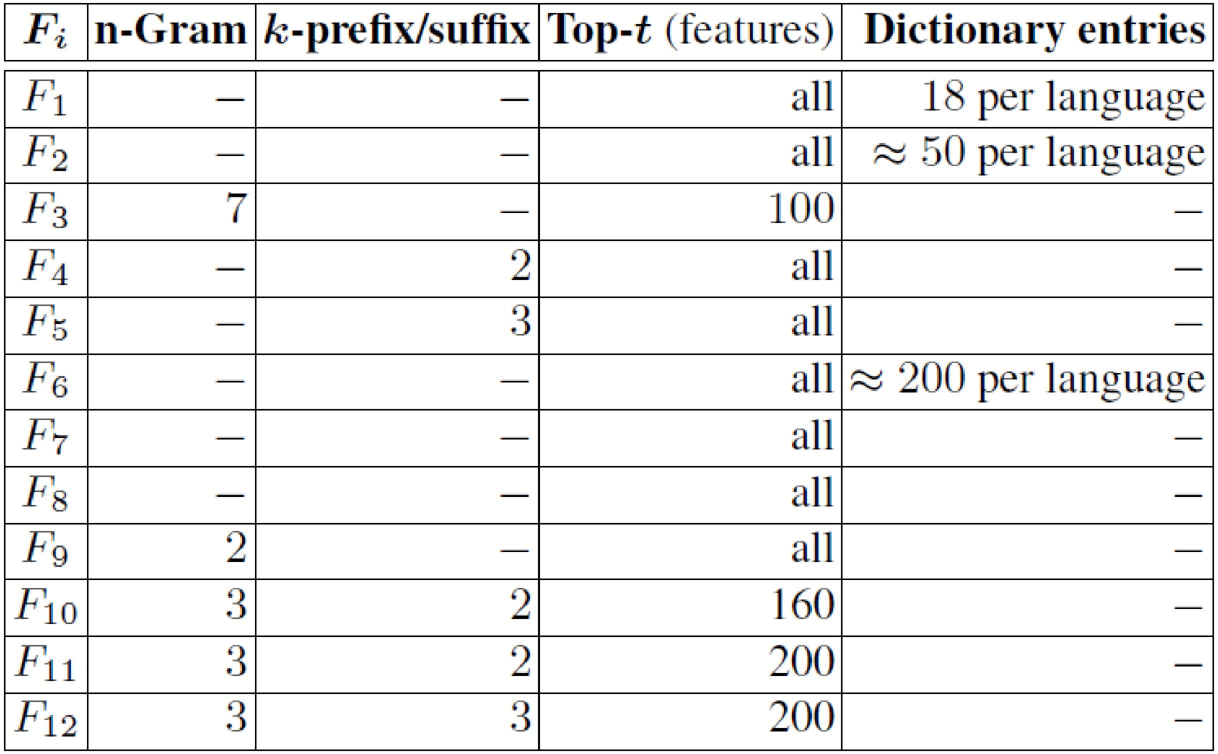
M.Sc. Inf.
Oren Halvani

Media Security and IT Forensics
Fraunhofer Institute for Secure Information Technology SIT

Rheinstrasse 75 | 64295 Darmstadt | Germany
Phone +49 6151 869-211 | Fax +49 6151 869-224
Mobile +49 179 2838686 | oren.halvani@sit.fraunhofer.de
www.sit.fraunhofer.de

USED PARAMETER-SETTINGS

- What kind of parameters were used for PAN and the german corpus…?



F_i	n-Gram	k-prefix/suffix	Top-t (features)	Dictionary entries
F_1	–	–	all	18 per language
F_2	–	–	all	≈ 50 per language
F_3	7	–	100	–
F_4	–	2	all	–
F_5	–	3	all	–
F_6	–	–	all	≈ 200 per language
F_7	–	–	all	–
F_8	–	–	all	–
F_9	2	–	all	–
F_{10}	3	2	160	–
F_{11}	3	2	200	–
F_{12}	3	3	200	–