Recognition of Whitehead-minimal elements in Free Groups of Large Ranks.

Alexei D. Miasnikov

The Graduate Center of CUNY
Computer Science
365 5th ave, New York, USA
amiasnikov@nyc.rr.com

Abstract. In this paper we introduce a pattern classification system to recognize words of minimal length in their automorphic orbits in free groups. This system is based on Support Vector Machines and does not use any particular results from group theory. The main advantage of the system is its stable performance in recognizing minimal elements in free groups with large ranks.

1 Introduction

This paper is a continuation of the work started in [5, 7]. In the previous papers we showed that pattern recognition techniques can be successfully used in abstract algebra and group theory in particular. The approach gives one an exploratory methods which could be helpful in revealing hidden mathematical structures and formulating rigorous mathematical hypotheses. Our philosophy here is that if irregular or non-random behavior has been observed during an experiment then there must be a pure mathematical reason behind this phenomenon, which can be uncovered by a proper statistical analysis.

In [7] we introduced a pattern recognition system that recognizes *minimal* (sometimes also called *Whitehead minimal*) words, i.e., words of minimal length in their automorphic orbits, in free groups. The corresponding probabilistic classification algorithm, a *classifier*, based on quadratic regression is very fast (linear time algorithm) and recognizes minimal words correctly with the high accuracy rate of more then 99%. However, the number of model parameters grows as a polynomial function of degree 4 on the rank of the free group. This limits applications of this system to free groups of small ranks (see Section 3.3).

In this paper we describe a probabilistic classification system to recognize Whitehead-minimal elements which is based on so-called *Support Vector Machines* [9, 10]. Experimental results described in the last section show that the system performs very well on different types of test data, including data generated in groups of large ranks.

The paper is structured as follows. In the next section we give a brief introduction to the Whitehead Minimization problem and discuss the limitations of the known deterministic procedure. In Section 3 we describe major components

of the classification system, including generation of training datasets and feature representation of elements in a free group. In the section we describe evaluation procedure and give empirical results on the performance of the system.

2 Whitehead's Minimization Problem

In this section we give a brief introduction to the Whitehead minimization problem.

Let X be a finite alphabet, $X^{-1} = \{x^{-1} \mid x \in X\}$ be the set of formal inverses of letters from X, and $X^{\pm 1} = X \cup X^{-1}$. For a word w in the alphabet $X^{\pm 1}$ by $|w|$ we denote the length of w. A word w is called *reduced* if it does not contain subwords of the type xx^{-1} or $x^{-1}x$ for $x \in X$. Applying reduction rules $xx^{-1} \to \varepsilon, x^{-1}x \to \varepsilon$ (where ε is the empty word) one can reduce each word w in the alphabet $X^{\pm 1}$ to a reduced word $\overline{w}$. The word $\overline{w}$ is uniquely defined and does not depend on the order in a particular sequence of reductions. The set $F = F(X)$ of all reduced words over $X^{\pm 1}$ forms a group with respect to multiplication defined by $u \cdot v = \overline{uv}$ (i.e., to compute the product of words $u, v \in F$ one has to concatenate them and then reduce). The group F with the multiplication defined as above is called a *free* group with *basis* X. The cardinality $|X|$ is called the *rank* of $F(X)$. Free groups play a central role in modern algebra and topology.

A bijection $\phi : F \to F$ is called an *automorphism* of F if $\phi(uv) = \phi(u)\phi(v)$ for every $u, v \in F$. The set $Aut(F)$ of all automorphisms of F forms a group with respect to composition of automorphisms. Every automorphism $\phi \in Aut(F)$ is completely determined by its images on elements from the basis X since $\phi(x_1 \dots x_n) = \phi(x_1) \dots \phi(x_n)$ and $\phi(x^{-1}) = \phi(x)^{-1}$ for any letters $x_i, x_i \in X^{\pm 1}$. An automorphism $t \in Aut(F(X))$ is called a *Whitehead's automorphism* if t satisfies one of the two conditions below:

1) t permutes elements in $X^{\pm 1}$;
2) t fixes a given element $a \in X^{\pm 1}$ and maps each element $x \in X^{\pm 1}, x \neq a^{\pm 1}$ to one of the elements x, xa, $a^{-1}x$, or $a^{-1}xa$.

By $\Omega(X)$ we denote the set of all Whitehead's automorphisms of $F(X)$. It is known [8] that every automorphism from $Aut(F)$ is a product of finitely many Whitehead's automorphisms.

The automorphic orbit $Orb(w)$ of a word $w \in F$ is the set of all automorphic images of w in F:

$$Orb(w) = \{v \in F \mid \exists \varphi \in Aut(F) \text{ such that } \varphi(w) = v\}.$$

A word $w \in F$ is called *minimal* (or *automorphically minimal*) if $|w| \leq |\varphi(w)|$ for any $\varphi \in Aut(F)$. By w_{min} we denote a word of minimal length in $Orb(w)$. Notice that w_{min} is not unique. By $WC(w)$ (the *Whitehead's complexity* of w) we denote a minimal number of automorphisms $t_1, \dots, t_m \in \Omega(X)$ such that $t_m \dots t_1(w) = w_{min}$. The algorithmic problem which requires finding w_{min} for a given $w \in F$

is called the *Minimization Problem* for F, it is one of the principal problems in combinatorial group theory and topology. There is a famous Whitehead's decision algorithm for the Minimization Problem, it is based on the following result due to Whitehead ([11]): if a word $w \in F(X)$ is not minimal then there exists an automorphism $t \in \Omega(X)$ such that $|t(w)| < |w|$. Unfortunately, its complexity depends on cardinality of $\Omega(X)$ which is exponential in the rank of $F(X)$. We refer to [6] for a detailed discussion on complexity of Whitehead's algorithms.

In this paper we focus on the *Recognition Problem* for minimal elements in F. It follows immediately from the Whitehead's result that $w \in F$ is minimal if and only if $|t(w)| \geq |w|$ for every $t \in \Omega(X)$ (such elements sometimes are called *Whitehead's minimal*). This gives one a simple deterministic decision algorithm for the Recognition Problem, which is of exponential time complexity in the rank of F. Note, that the worst case in terms of the rank occur when the input word w is already minimal. In this situation all of the Whitehead automorphisms $\Omega(X)$ have to be applied.

Construction of a probabilistic classifier which recognizes words of minimal length allows one to solve the recognition problem quickly in expense of a small classification error. Such classifier can be used as a fast minimality check heuristic in a deterministic algorithm which solves the minimization problem.

It is convenient to consider the Minimization Problem only for cyclically reduced words in F. A word $w = x_1 \dots x_n \in F(X)$ ($x_i \in X^{\pm 1}$) is *cyclically reduced* if $x_1 \neq x_n^{-1}$. Clearly, every $w \in F$ can be presented in the form $w = u^{-1}\widetilde{w}u$ for some $u \in F(X)$ and a cyclically reduced element $\widetilde{w} \in F(X)$ such that $|w| = |\widetilde{w}| + 2|u|$. This $\widetilde{w}$ is unique and it is called a *cyclically reduced form* of w. Every minimal word in F is cyclically reduced, therefore, it suffices to construct a classifier only for cyclically reduced words in F.

3 Recognition of minimal words in free groups

One of the main applications of Pattern Recognition techniques is classification of a variety of given objects into categories. Usually classification algorithms or *classifiers* use a set of measurements (properties, characteristics) of objects, called *features*, which gives a descriptive representation for the objects. We refer to [2] for detailed introduction to pattern recognition techniques.

In this section we describe a particular pattern recognition system PR_{MIN} for recognizing minimal elements in free groups. The corresponding classifier is a supervised learning classifier which means that the decision algorithm is "trained" on a prearranged dataset, called *training* dataset in which each pattern is labelled with its true class label. The algorithm is based on Support Vector Machines (SVM) classification algorithm.

In Section 1 we have stressed that the number of parameters required to be estimated by the classification model based on quadratic regression is of order $\mathrm{O}(n^4)$, where n is the rank of a free group F_n. This constitutes two main problems. First, in order to compute the parameters we have to multiply and

decompose matrices of size equal to the number of the coefficients itself. For large n, the straightforward computation of such matrices might be impossible due to the memory size restrictions. Another problem, which is perhaps the major problem, is due to the fact that the number of observations in the training set needs to be about 100 times more than the number of the coefficients to be estimated. When n is large (for $n = 10$ the required number of observations is about 14,440,000) it is a significant practical limitation, especially when the data generation is time consuming.

One of the main attractive features of the Support Vector Machines is their ability to employ non-linear mapping without essential increase in the number of parameters to be estimated and, therefore, in computation time.

3.1 Data generation: training datasets

A pseudo-random element w of $F(X)$ can be generated as a pseudo-random sequence $y_1, \dots, y_l$ of elements $y_i \in X^{\pm 1}$ such that $y_i \neq y_{i+1}^{-1}$, where the length l is also chosen pseudo-randomly. However, it has been shown in [4] that randomly taken cyclically reduced words in F are already minimal with asymptotic probability 1. Therefore, a set of randomly generated cyclic words in F would be highly biased toward the class of minimal elements. To obtain fair training datasets we use the following procedure.

For each positive integer $l = 1, \dots, L$ we generate pseudo-randomly and uniformly K cyclically reduced words from $F(X)$ of length l. Parameters L and K were chosen to be 1000 and 10 for pure practical reasons. Denote the resulting set by W. Then using the deterministic Whitehead algorithm we construct the corresponding set of minimal elements

$$W_{min} = \{w_{min} \mid w \in W\}.$$

With probability 0.5 we substitute each $v \in W_{min}$ with the word $\widetilde{t(v)}$, where t is a randomly and uniformly chosen automorphism from $\Omega(X)$ such that $|\widetilde{t(v)}| > |v|$ (if $|\widetilde{t(v)}| = |v|$ we chose another $t \in \Omega(X)$, and so on). Now, the resulting set L is a set of pseudo-randomly generated cyclically reduced words representing the classes of minimal and non-minimal elements in approximately equal proportions. It follows from the construction that our choice of non-minimal elements w is not quite representative, since all these elements have Whitehead's complexity one (which is not the case in general). One may try to replace the automorphism t above by a random finite sequence of automorphisms from Ω to get a more representative training set. However, we will see in Section 4 that the training dataset L is sufficiently good already, so we elected to keep it as it is.

From the construction we know for each element $v \in L$ whether it is minimal or not. Finally, we create a training set

$$D = \{< v, P(v) > \mid v \in L\},$$

where

$$P(v) = \begin{cases} 1, \, v \text{ is minimal;} \\ 0, \text{ otherwise.} \end{cases}$$

3.2 Features

To describe the feature representation of elements from a free group $F(X)$ we need the following

Definition 1. *Labelled Whitehead Graph $WG(v) = (V, E)$ of an element $v \in F(X)$ is a weighted non-oriented graph, where the set of vertices V is equal to the set $X^{\pm 1}$, and for $x_i, x_j \in X^{\pm 1}$ there is an edge $(x_i, x_j) \in E$ if the subword $x_i x_j^{-1}$ (or $x_j x_i^{-1}$) occurs in the word v viewed as a cyclic word. Every edge (x_i, x_j) is assigned a weight l_{ij} which is the number of times the subwords $x_i x_j^{-1}$ and $x_j x_i^{-1}$ occur in v.*

Whitehead Graph is one of the main tools in exploring automorphic properties of elements in a free group [4, 8].

Now, let $w \in F(X)$ be a cyclically reduced word. We define features of element w as follows. Let $l(w)$ be a vector of edge weights in the Whitehead Graph $WG(w)$ with respect to a fixed order. We define a feature vector $f(w)$ by

$$f(w) = \frac{1}{|w|} l(w).$$

This is the basic feature vector in all our considerations.

3.3 Decision Rule

Below we give a brief description of the classification rule based on Support Vector Machine.

Let $D = \{w_1, \ldots, w_N\}$, $w \in F(X)$ be a training set and $D' = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i = f(w_i)$ be the set of feature vectors with the corresponding labels $y_1, \ldots, y_N$, where

$$y_i = \begin{cases} +1, \text{ if } P(w_i) = 1; \\ -1, \text{ otherwise.} \end{cases}$$

Definition 2. *The margin of an example $(\mathbf{x}_i, y_i)$ with respect to a hyperplane $(\mathbf{w}, b)$ defined as the quantity*

$$\gamma_i = y_i(\mathbf{w}' \cdot \mathbf{x} + b).$$

Note that $\gamma_i > 0$ corresponds to the correct classification of $(\mathbf{x}_i, y_i)$.

Let $\gamma_+ (\gamma_-)$ be the smallest margin among all positive (negative) points. Define the margin of separation

$$\gamma = \gamma_+ + \gamma_-.$$

A Support Vector Machine (SVM) is a statistical classifier that attempts to construct a decision hyperplane $(\mathbf{w}, b)$ in such a way that the margin of separation γ between positive and negative examples is maximized [9, 10].

We wish to find a hyper-plane which will separate the two classes such that all points on one side of the hyper-plane will be labelled +1, all points on the other side will be labelled -1. Define a discriminant function

$$g(\mathbf{x}) = \mathbf{w}^{*\prime} \cdot \mathbf{x} + b^*,$$

where $\mathbf{w}^*, b^*$ are the parameters of the optimal hyper-plane. Function $g(\mathbf{x})$ gives the distance from an arbitrary $\mathbf{x}$ to the optimal hyper-plane.

Parameters of the optimal hyperplane are obtained by maximizing the margin, which is equivalent to minimizing the cost function

$$\Phi(\mathbf{w}) = \|\mathbf{w}\|^2 = \mathbf{w}' \cdot \mathbf{w},$$

subject to the constraint that

$$y_i \left(\mathbf{w}' \cdot \mathbf{x}_i + b\right) - 1 \geq 0, \ i = 1, \ldots, N.$$

This is an optimization problem with inequality constraints and can be solved by means of Lagrange multipliers. We form the Lagrangian

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}' \cdot \mathbf{w} - \sum_{i=1}^{N} \alpha_i \left[y_i \left(\mathbf{w}' \cdot \mathbf{x}_i + b\right) - 1\right],$$

where $\alpha_i \geq 0$ are the Lagrange multipliers. We need to minimize $L(\mathbf{w}, b, \alpha)$ with respect to $\mathbf{w}$, b while requiring that derivatives of $L(\mathbf{w}, b, \alpha)$ with respect to all the α_i vanish, subject to the constraint that $\alpha_i \geq 0$. After solving the optimization problem the discriminant function

$$g(x) = \sum_{i=1}^{N} y_i \alpha_i^* \mathbf{x}_i' \cdot \mathbf{x} + b^*.$$

where α_i^*, b^* are the parameters of the optimal decision hyperplane. It shows that the distance can be computed as a weighted sum of the training data and the Lagrange multipliers, and that the training vectors $\mathbf{x}_i$ are only used in inner products.

One can extend linear case to non-linearly separable data by introducing a kernel function

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j),$$

where $\varphi(\mathbf{x})$ is some non-linear mapping into (possibly infinite) space H,

$$\varphi : \mathbb{R}^n \longmapsto H.$$

Since Support Vector Machines use only inner products to compute the discriminant function, given kernel $K(x_i, x_j)$, we can train a SVM without ever having to know $\varphi(x)$ [3]. The implication of this is that the number of parameters that has to be learned by the SVM does not depend on the choice of the kernel and, therefore, mapping φ. This gives an obvious computational advantage when mapping the original feature space into a higher dimensional space which is the main obstacle in the previous approach based on quadratic regression.

Examples of typical kernel functions are:

– Linear :

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i' \cdot \mathbf{x}_j$$

– Polynomial:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i' \cdot \mathbf{x}_j)^d$$

– Exponential:

$$K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}$$

– Neural Networks:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh\left(\theta_1 \mathbf{x}_i' \cdot \mathbf{x}_j - \theta_2\right)$$

Now we can define the decision rule used by the system. The classification algorithm has to predict the value $P(w)$ of the predicate P for a given word w. The corresponding decision rule is

$$\text{Decide } P(w) = \begin{cases} 1, \text{ if } g(f(w)) \geq 0; \\ 0, \text{ otherwise.} \end{cases}$$

4 Evaluation of the system

4.1 Test datasets

To test and evaluate our pattern recognition system we generate several test datasets of different types:

– A test set S_e which is generated by the same procedure as for the training set D, but independently of D.
– A test set S_R of pseudo-randomly generated cyclically reduced elements of $F(X)$, as described in Section 3.1.
– A test set S_P of pseudo-randomly generated cyclically reduced *primitive* elements in $F(X)$. Recall that $w \in F(X)$ is primitive if and only if there exists a sequence of Whitehead automorphisms $t_1 \dots t_m \in \Omega(X)$ such that $t_m \dots t_1(x) = w$ for some $x \in X^{\pm 1}$. Elements in S_P are generated by the procedure described in [6], which, roughly speaking, amounts to a random choice of $x \in X^{\pm 1}$ and a random choice of a sequence of automorphisms $t_1 \dots t_m \in \Omega(X)$.
– A test set S_{10} which is generated in a way similar to the procedure used to generate the training set D. The only difference is that the non-minimal elements are obtained by applying not one, but several randomly chosen automorphisms from $\Omega(X)$. The number of such automorphisms is chosen uniformly randomly from the set $\{1, \dots, 10\}$, hence the name.

For more details on the generating procedure see [6].

To show that performance of Support Vector Machines is acceptable for free groups, including groups of large ranks, we run experiments with groups of ranks 3,5,10,15,20. For each group we construct the training set D and test sets S_e, S_{10}, S_R, S_P using procedures described previously. Some statistics of the datasets are given in Table 1.

4.2 Accuracy measure

To evaluate the performance of the classification system PR_{MIN} we define an accuracy measure A.

Let D_{eval} be a test data set and

$$K = |\{w \mid decide(w) = P(w),\ w \in D_{eval}\}|$$

be the number of correctly classified elements in D_{eval}. To evaluate the performance of a given pattern classification system we use a simple accuracy measure:

$$A = \frac{K}{|D_{eval}|},$$

which gives the fraction of the correctly classified elements from the test set D_{eval}.

Notice, that the numbers of correctly classified elements follow the Binomial distribution and A can be viewed as an estimate of probability p of a word being classified correctly.

We are interested in constructing a confidence interval for probability p. For binomial variates, exact confidence intervals do not exist in general. One can obtain an approximate $100(1-\alpha)\%$ confidence interval $[p_S, p_L]$ by solving the following equations for p_S and p_L:

$$\sum_{i=0}^{K} \binom{|D_{eval}|}{i} p_S^i (1-p_S)^{|D_{eval}|-i} = \alpha/2,$$

$$\sum_{i=K}^{|D_{eval}|} \binom{|D_{eval}|}{i} p_L^i (1-p_L)^{|D_{eval}|-i} = \alpha/2$$

for a given α.

Exact solutions to the equations above can be obtained by re-expressing in terms of the incomplete beta function (see [1] for details).

4.3 Results of experiments

Experiments were repeated with the following types of kernel functions:

K^1: linear;
K^2: quadratic $(b\mathbf{x}_i' \cdot \mathbf{x}_j + c)^2$;
K^3: polynomial of degree 3 $(b\mathbf{x_i}' \cdot \mathbf{x_j} + c)^3$;
K^4: polynomial of degree 4 $(b\mathbf{x_i}' \cdot \mathbf{x_j} + c)^4$;
K^e: Gaussian $e^{-\gamma||\mathbf{x_i}-\mathbf{x_j}||^2}$,

where $\mathbf{x_i}$, $\mathbf{x_j}$ are the feature vectors obtained with mapping f_{WG} and $\mathbf{x_i} \cdot \mathbf{x_j}$ is the inner product of $\mathbf{x_i}$ and $\mathbf{x_j}$.

The results of the experiments presented in Table 2. It shows that SVM with appropriate kernel perform well not only on free groups of small ranks but on

groups of large ranks as well. The experiments confirmed observations, made previously, that classes of minimal and non-minimal words are not linearly separable. Moreover, once the rank and, therefore dimensionality of the feature space grows, quadratic mapping does not guarantee the high classification accuracy. As one might expect, the accuracy increases when the degree of the polynomial mapping increases. Nevertheless, even with the polynomial kernel K^4 of the degree $d = 4$, Support Vector Machine is not able to perform accurate classification in groups F_{15} and F_{20}. However, Gaussian kernel produces stable and accurate results for all test datasets, including sets of elements in free groups of large ranks. This indicates that points in one of the classes (minimal or non-minimal) are compactly distributed in the feature space and can be accurately described as a Gaussian. We also can observe that Gaussian representation can be applied to only one of the classes. If the opposite was true, then the problem of separating the two classes would be much simpler and at least the quadratic mapping should have been as accurate as K^e.

We conclude this section with the following conclusions:

1. With appropriate kernel function Support Vector Machines approach performs very well in the task of classification of Whitehead-minimal words in free groups of various ranks, including groups of large ranks.
2. The best over all results are obtained with the Gaussian kernel K^e. This indicates that one of the classes is compact and can be bounded by a Gaussian function.
3. Regression approach is still would be preferable for groups of small ranks due to its simplicity and smaller resource requirements. However, the SMVs should be used for groups of larger ranks where the size of the training sets required to perform regression with non-linear preprocessing mapping becomes practically intractable.

References

1. Milton Abramowitz and Irene Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Table.* Dover Publications, Inc., 1972.
2. R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley-Interscience, 2nd edition, 2000.
3. S. Haykin. *Neural networks: A comprehensive foundation.* Prentice Hall, Upper Sadle River, NJ, 1999.
4. I. Kapovich, P. Schupp, and V. Shpilrain. Generic properties of whitehead's algorithm, stabilizers in $aut(f_k)$and one-relator groups. Preprint, 2003.
5. A.D. Miasnikov and R.M. Haralick. Regression analysis and automorphic orbits in free groups of Rank 2. 17th International Conference on Pattern Recognition. To appear, 2004.
6. A.D. Miasnikov and A.G. Myasnikov. Whitehead method and genetic algorithms. Contemporary Mathematics, 349:89-114, 2004.
7. R.M. Haralick, A.D. Miasnikov and A.G. Myasnikov. Pattern Recognition Approaches to Solving Combinatorial Problems in Free Groups. Contemporary Mathematics, 349:197-213, 2004.

8. L. R. and P. Schupp. *Combinatorial Group Theory*, volume 89 of *Series of Modern Studies in Math*. Springer-Verlag, 1977.
9. V.N. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.
10. V.N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, New York, 2000.
11. J. H. C. Whitehead. On equivalent sets of elements in a free group. *Annals of Mathematic*, 37(4):782-800, 1936.

Dataset	size	% min	% non-min	(min,avg,max) word lengths
D	20000	49.1	50.9	(3,558.2,1306)
S_e	5000	48.9	51.1	(3,559,1292)
S_{10}	5000	49.1	50.9	(3,1016.5,13381)
S_R	5000	98.3	1.7	(3,501.2,999)
S_P	3850	0.0	100.0	(3,194.7,8719)

a) F_3;

Dataset	size	% min	% non-min	(min,avg,max) word lengths
D	20000	48.5	51.5	(5,581.3,1388)
S_e	5000	49.2	50.8	(8,583.7,1382)
S_{10}	5000	48.0	52.0	(7,1693.22,28278)
S_R	5000	97.2	2.8	(6,504.2,999)
S_P	2900	0.0	100.0	(5,656.9,22430)

c) F_5;

Dataset	size	% min	% non-min	(min,avg,max) word lengths
D	9660	48.9	51.1	(26,617.4,1461)
S_e	4811	49.2	50.8	(26,619.7,1443)
S_{10}	4837	49.5	50.5	(29,2589.8,65274)
S_R	4867	96.5	3.5	(18,512.7,999)
S_P	165	0.0	100.0	(12,150.8,1459)

a) F_{10};

Dataset	size	% min	% non-min	(min,avg,max) word lengths
D	9357	49.5	50.5	(41,635.3,1472)
S_e	4685	49.2	50.8	(40,642.5,1462)
S_{10}	4722	49.7	50.3	(46,3056.6,53422)
S_R	4755	95.3	4.7	(26,523.8,999)
S_P	870	0.0	100.0	(28,1109.3,4981)

b) F_{15};

Dataset	size	% min	% non-min	(min,avg,max) word lengths
D	9144	49.6	50.4	(47,658.3,1488)
S_e	4576	49.3	50.7	(48,659.8,1484)
S_{10}	4597	49.1	50.9	(64,3351.4,68316)
S_R	4643	94.0	6.0	(48,534.9,999)
S_P	182	0.0	100.0	(66,945.1,4762)

c) F_{20};

Table 1. Description of the training and test datasets in free groups F_{10}, F_{15} and F_{20}.

	All elements					Elements with $\|w\| > 100$				
Kernel	F_3	F_5	F_{10}	F_{15}	F_{20}	F_3	F_5	F_{10}	F_{15}	F_{20}
K^1	.844	.805	.729	.676	.644	.859	.810	.738	.680	.648
K^2	.995	.978	.881	.782	.710	.999	.977	.880	.792	.711
K^3	.996	.988	.962	.888	.772	1.00	.996	.968	.894	.773
K^4	.996	.989	.984	.951	.832	1.00	.997	.991	.956	.834
K^e	.995	.986	.982	.988	.990	1.00	.999	.999	.998	.995

a) accuracy evaluated on the set S_e;

	All elements					Elements with $\|w\| > 100$				
Kernel	F_3	F_5	F_{10}	F_{15}	F_{20}	F_3	F_5	F_{10}	F_{15}	F_{20}
K^1	.806	.783	.760	.670	.676	.818	.798	.768	.717	.687
K^2	.993	.988	.885	.811	.750	.997	.971	.893	.814	.751
K^3	.994	.993	.969	.893	.809	1.00	.998	.976	.897	.810
K^4	.995	.993	.993	.954	.866	1.00	.998	.997	.957	.870
K^e	.995	.993	.985	.986	.989	1.00	1.00	.999	.999	.994

b) accuracy evaluated on the set S_{10};

	All elements					Elements with $\|w\| > 100$				
Kernel	F_3	F_5	F_{10}	F_{15}	F_{20}	F_3	F_5	F_{10}	F_{15}	F_{20}
K^1	.880	.833	.743	.710	.691	.915	.887	.768	.729	.727
K^2	.990	.986	.890	.812	.754	.999	.969	.903	.830	.790
K^3	.991	.991	.961	.893	.824	1.00	.996	.985	.911	.842
K^4	.992	.991	.973	.940	.867	1.00	.997	.999	.970	.883
K^e	.990	.986	.980	.973	.970	1.00	1.00	.999	.989	.973

c) accuracy evaluated on the set S_R;

	All elements					Elements with $\|w\| > 100$				
Kernel	F_3	F_5	F_{10}	F_{15}	F_{20}	F_3	F_5	F_{10}	F_{15}	F_{20}
K^1	.732	.798	.770	.694	.610	.674	.682	.769	.690	.612
K^2	.999	.997	.824	.722	.610	.993	.993	.785	.723	.626
K^3	1.00	1.00	.915	.777	.632	1.00	1.00	.877	.756	.648
K^4	1.00	1.00	.982	.821	.659	1.00	1.00	.985	.816	.665
K^e	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

d) accuracy evaluated on the set S_P.

Table 2. Performance of the Support Vector Machine classifier in free groups F_3, F_5, F_{10}, F_{15} and F_{20}.