Schedae Informaticae Vol. 24 (2015): 9–19
doi: 10.4467/20838476SI.15.001.3023

Analysis of Compounds Activity Concept Learned by SVM Using Robust Jaccard Based Low-dimensional Embedding

Stanisław Jastrzębski, Wojciech Marian Czarnecki
Faculty of Mathematics and Computer Science
Jagiellonian University
ul. Łojasiewicza 6, 30-348 Kraków, Poland
e-mail: {*stanislaw.jastrzebski, wojciech.czarnecki*}*@uj.edu.pl*

Abstract. Support Vector Machines (SVM) with RBF kernel is one of the most successful models in machine learning based compounds biological activity prediction. Unfortunately, existing datasets are highly skewed and hard to analyze. During our research we try to answer the question how deep is activity concept modeled by SVM. We perform analysis using a model which embeds compounds' representations in a low-dimensional real space using near neighbour search with Jaccard similarity. As a result we show that concepts learned by SVM is not much more complex than slightly richer nearest neighbours search. As an additional result, we propose a classification technique, based on Locally Sensitive Hashing approximating the Jaccard similarity through minhashing technique, which performs well on 80 tested datasets (consisting of 10 proteins with 8 different representations) while in the same time allows fast classification and efficient online training.

Keywords: Support Vector Machines, Locally Sensitive Hashing, Jaccard similarity.

1. Introduction

One of the most popular machine larning methods in ligand-based virtual screening [1] (process of automated selection of chemical compounds being new drug candidates) is Vapnik's Support Vector Machine (SVM [2]). Despite its good numerical results relatively small amount of attention has been paid to actual activity concepts modeled

by this method when using the RBF kernel. In this paper we try to answer the following question:

> *Is Support Vector Machine with RBF kernel learning any complex data patterns exploiting compound activity or does it degenerate to nearly nearest neighbour search?*

There are at least few reasons to raise such a question. Chemical compounds activity datasets are gathered in a highly biased way which heavily violates the i.i.d. assumption of machine learning models. These datasets are mostly obtained through analysis of published results of pharmacologists who do not publish negative results unless inactive compound is extremely similar to some known active one. Similarly, due to the bias on the already known drugs, enormous parts of input space are completely not investigated. In order for a machine learning model to be effective in the process of automated drug design it should be able to model the concept of activity which can be then applied to underresearched parts of the molecules space. However, if current models actually degenerate to the near neighbour search, then their fitness for chemical applications is very limited.

In this paper we show that RBF based SVM, which achieves high evaluation scores (between 80% and 100% of balanced accuracy) is actually a slightly enriched nearest neighbour model. We do this by constructing an efficient, low dimensional embedding of chemical datasets in real space, where linear models can only base their decision on a very limited, local neighbourhood of each molecule. Through evaluation on many proteins and fingerprints we show that obtained results are very similar to the ones obtained by SVM with RBF kernel.

As an additional result our low-dimensional embedding followed by the linear classifier appears to be a robust model for such a problem, which has much smaller classficiation complexity than kernel SVMs and can be efficiently trained in an online scenario due to the utilization of locality sensitive hashing and logistic regression.

2. Methods

In order to show that RBF based SVM learns very simple concepts of compounds activity we will construct an extremely simplified model which embeds each point in the 8-dimensional real space and build a linear model on such a representation. Our main goal is to show that such simple classifier performs nearly identical to the state of the art method (which supports our claim) and as a side result – we show that it might be a very effective (in terms of classification time and future online learning) alternative.

We start with some basic description of RBF based SVM and then introduce Localy Sensitive Hashing, a procedure which makes our embedding computationally feasible, finally we describe whole method and discuss its properties.

2.1. Support vector machines

Support Vector Machines (SVM), model proposed by Vapnik et al. [2], is one of the most successful classifiers of the last decade mostly thanks to two characteristics: first, its well motivated regularization method in the linear case, second, its quite efficient delinearization. Further is achieved using so called kernel trick, where we substitute each scalar product between elements of the input space with the appropriate kernel function which is simply a scalar product in some (potentially much richer) space. One of the most commonly used kernels (due to its simplicity and general applicability) is a Gaussian (RBF) kernel

$$K(x_i, x_j) = \exp\left(\gamma \|x_i - x_j\|^2\right) = A\left\langle \mathcal{N}\left(x_i, \tfrac{1}{(2\gamma)^2}\right), \mathcal{N}\left(x_j, \tfrac{1}{(2\gamma)^2}\right)\right\rangle \quad (1)$$

$$= A\int_{\mathbb{R}^d} \mathcal{N}\left(x_i, \tfrac{1}{(2\gamma)^2}\right)[x] \cdot \mathcal{N}\left(x_j, \tfrac{1}{(2\gamma)^2}\right)[x]dx \quad (2)$$

for an appropriate normalizing constant A and $\mathcal{N}(m, \sigma^2)$ denoting the normal probability distribution function with mean m and variance σ^2. Such an approach has numerous theoretical advantages [3], however also has one big flaw, it increases training time up to $\mathcal{O}(N^3)$ and classification time to $\mathcal{O}(N)$. It is worth noting, that kernel trick is much deeper modification than simply transforming the input dataset X to $K(X, X)$ and runing a linear SVM on such N-dimensional data, reconstruction of the underlying Reproducing Hilbert Kernel Space (RHKS, feature space) is much more complex task [4]. In particular as one can see from Eq. 1 RHKS consists of Gaussian mixture models leading to infinitely dimensional feature space. Such complexity becomes impossible to use once training set size grows over few thousands (especially concerning that we need to fit two hyperparamters). One of the possible approaches to reduction of training time is direct construction of the feature space projection which does not require kernel trick to create non-linear classification in the input space. While there are some existing methods (including so called Nystroem kernel approximation [5]) we will show very simple approach which can achieve comparative results with SVM RBF while in the same time maps input space to $\mathbb{R}^8$, making training procedure extremely robust.

2.2. Low-dimensional near neighbour embedding

The core idea behind our embedding is to encode each data point based only on its very limited neighbourhood. This way any classifier trained on such data can not construct any global concept of compounds activity. We propose to analyze k-nearest neighbours of a given points. We could simply concatenate the representations of these neighbours to create a new representation of our point however this would lead to an explosion of problem's dimensionality (kd for data in $\mathbb{R}^d$) and resulting classification problem might be even harder than the original one. Instead, we propose to simply take some basic similarity based statistics of this neighbourhood, namely we analyze

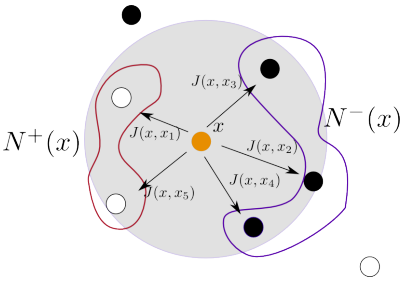
number of samples of each class and their mean, minimum and maximum similarity under some measure. This way we get 8 dimensional representation as a result of non-linear transformation of the input space. Let us put this in a more formal way.

Definition 1 (Local Statistics Embedding) *For a given dataset and arbitrary similarity measure S we define a Local Statistics Embedding (LSE) as*

$$\tau_k(x) = [\quad \begin{matrix} |N^-(x)| & |N^+(x)| & \text{mean } S(N^-(x)) & \text{mean } S(N^+(x)) \\ \min S(N^-(x)) & \min S(N^+(x)) & \max S(N^-(x)) & \max S(N^+(x)) \end{matrix} \quad]^T,$$

where $N^l(x)$ is a sequence of samples with label l of the k nearest neighbours of x in terms of S.

Figure 1 shows an example embedding of a point with $k = 5$ and using some similarity denoted as J, positive samples are white and negative are black.



$$\tau_5(x) = \begin{bmatrix} 3 \\ 2 \\ \frac{J(x,x_1)+J(x,x_2)+J(x,x_4)}{3} \\ \frac{J(x,x_3)+J(x,x_5)}{2} \\ \min\{J(x,x_3), J(x,x_5)\} \\ \min\{J(x,x_1), J(x,x_2), J(x,x_4)\} \\ \max\{J(x,x_3), J(x,x_5)\} \\ \max\{J(x,x_1), J(x,x_2), J(x,x_4)\} \end{bmatrix}$$

Figure 1. LSE embedding visualisation.

It is worth noting that training a linear model on LSE generalizes a kNN approach in a natural way. Following observation makes it more clear.

Observation 1 *Class of hypotheses representable by LSE embedding followed by a linear model is a strict superset of the kNN hypotheses space as well as similarity weighted kNN hypotheses space.*

In general, kNN search for an arbitrary similarity measure requires $\mathcal{O}(N)$ operations for each query, so in particular embedding of the dataset takes $\mathcal{O}(N^2)$ and classification is $\mathcal{O}(N)$ which resembles the SVM RBF complexity of classification. Fortunately exploiting LSH technique we can reduce this time to sublinear per query, leading to sublinear data embedding followed by training of the linear model on low-dimensional space which is linear itself [6]. Assuming that LSH query complexity is $\mathcal{O}(N^p)$ [7] (where $p = \frac{\log(1/p_1)}{\log(1/p_2)}$ and p_1 is a lower bound on the probability that two points within neighbourhood hash to the same bucket and p_2 is the upper bound of the probability that two points outside this neighbourhood hash to the same bucket) we get a training time of $\mathcal{O}(N^{1+p})$ and classification time of $\mathcal{O}(N^p)$.

One of the interesting properties of this classifier is the fact that we can efficiently train this model in an online scenario. Given built LSH and trained linear model (for example Logistic Regression) and given new point (x, y) which should be added to the

train set we can simply update existing LSH by querying $2k$ nearest points and we update only those points. Then we can train our linear classifier in an online fashion, as the solution should not change much, this operation is also sublinear [8].

2.3. Localy Sensitive Hashing for LSE and Jaccard similarity

Localy Sensitive Hashing (LSH [7]) technique is an approximate technique for building a data structure able to answer near-neighbourhood querying in sublinear time. Its basic idea is to group points in buckets using hashing techniques in such a way that they approximate known metrics. In particular one can use random hyperplanes hash functions to build cosine similarity LSH, or other function for Euclidean metric. In our case we use MinHash [7] functions which approximate the Jaccard similarity[1] as it is a good measure for compounds (represented with binary fingerprints) similarity [9].

In a typical application of LSH we query for points that are similar to given x above some predefined threshold of interest. In our case we are interested in retrieving a k element neighbourhood of a compound which is impossible to answer using single basic LSH if data is non-uniformly distributed. To support such a query we constructed multiple LSHs with sequentially adjusted thresholds and query thenm in a descending order. In other words we query for very similar points, and if there are less than $2k$ such results we broaden our query until at least $2k$ neighbours are found, from which we pick k nearest neighbours.

3. Evaluation

Our problem consists of predicting compound activity so it is a binary classification task. One of the main issues is choosing appropriate representation. We evaluated our models on 8 popular fingerprints [10]: EstateFP, ExtFP, KlekFP, KlekFPCount, MACCSFP, PubChemFP, SubFP, SubFPCount. Each fingerprint is used to predict compound's activity of 10 proteins: 5-HT_7, 5-HT_6, SERT, 5-HT_{2C}, 5-HT_{2A}, HIVi, H_1, hERG, Cathepsin, M_1. Each fingerprint–protein pair is a separate experiment for which we evaluate all of the models. This gives us significant statistical robustness of our conclusions.

Fingerprints differ significantly in terms of sparsity, length and existence of non-binary values. The shortest fingerprint is EstateFP (with length of 36) and the longests are KlekFP and KlekFPCount (with length of 1452). KlekFP and KlekFPCount are also the most sparse fingerprints.

For each LSE based model raw dataset is converted to a binary representation (by binning values into constant length bins). Then sequential LSH is trained on the

[1] For two sets A and B Jaccard similarity is defined as $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$.

whole dataset and LSE embedding is constructed. Figure 2 presents kernel density estimation of LSE features distribution per class for 5-HT$_{2C}$ protein with MACCSFP fingerprint.

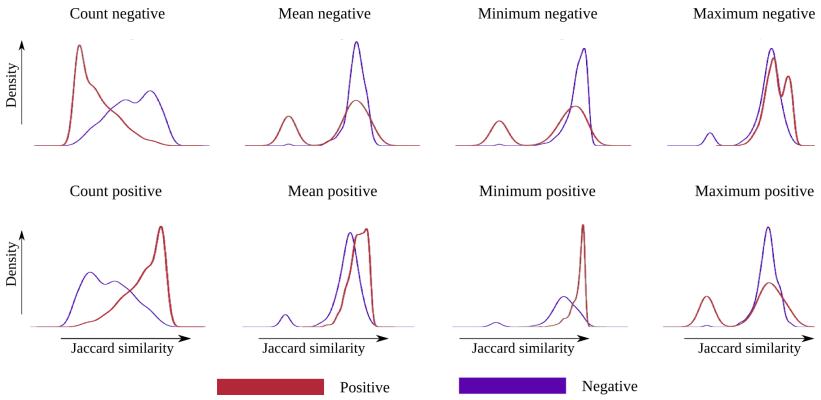


Figure 2. Comparision of distributions of LSE parameters.

As one can see, even separation based on projections on each axis might lead to a good discrimination, for example minimum similarity to the positive samples is a good discriminative measure, in fact SVM bases 45% of all weights on this particular feature when trained on this protein-fingerprint pair (and 40% on average across all protein-fingerprint pairs).

We used seven models during this evaluation, namely: kNN (with Jaccard similarity, denoted as kNN), kNN using LSH for querying for neareast points (denoted as LSH+kNN), SVM with RBF kernel (denoted as SVM RBF), SVM with Jaccard kernel (denoted as SVM Jaccard), proposed embedding followed by linear SVM (denoted as LSE+SVM) and proposed embedding followed by Logistic Regression (denoted as LSE+LR). All methods were implemented in Python with the use of numpy, scipy and scikit-learn. All hyperparameters of SVM RBF (C and γ), rest of the SVMs (C and k of LSE), LR (C) and KNN (k) were fitted using grid search technique. C was looked in the range $10^{-5} \dots 10^{6}$, γ was looked in the range $10^{-14} \dots 10^{0}$ and LSE K values 15, 20 and 30 were checked. LSH used 1000 MinHash functions and sequential cascade consisted of 10 LSH models. Each model was trained with a time-out, however for SVM RBF we trained it also without the time-out to make sure that it can achieve its best results.

Evaluation was performed in 10-fold stratified cross validation fashion under the balanced accuracy (BAC) metric[2] which is a popular balanced metric for binary classification [11].

Table 1 consists of summarization of results obtained by each model. We measure amount of experiments in which given method achieved better BAC than SVM RBF method. We also report how big is this difference on average.

One should notice that on 5 out of 8 fingerprints SVM Jaccard is significantly better than SVM RBF and our models based on proposed embedding are often even

[2] $\mathrm{BAC} = \frac{1}{2}\left(\frac{\mathrm{TP}}{\mathrm{TP+FN}} + \frac{\mathrm{TN}}{\mathrm{TN+FP}}\right)$.

stronger (even on datasets where SVM Jaccard performs well). Second, our models are better than SVM RBF in most cases. They perform slightly worse on long fingerprints which should be subject for further research. Finally, surprising strength of LSE based models is supporting our claim that SVM RBF doesn't learn any deep concepts but rather degenerates to the nearest neighbour search. This is probably the result of the fact that analyzed type of data violates the basic ML assumption of i.i.d. data generation.

Table 1. Comparison of how many times given model achieved better BAC than SVM RBF. For each model fraction of times it is better than the baseline model SVM RBF and average BAC score loss is reported.

Fingerprint	kNN (%)		LSE + kNN (%)		RBF Nystroem (%)	
EstateFP	20	−2.5	0	−6.8	0	−3.5
ExtFP	0	−2.0	0	−2.6	0	−5.5
KlekFP	0	−2.5	0	−3.4	0	−5.6
KlekFPCount	10	−1.2	0	−2.0	0	−4.8
MACCSFP	10	−0.8	0	−1.7	0	−4.8
PubChemFP	0	−2.9	0	−3.7	0	−5.7
SubFP	50	−0.8	20	−2.4	0	−3.8
SubFPCount	0	−3.3	60	−0.6	0	−4.1
Best	0	−2.4	0	−2.9	0	−4.9
Fingerprint	LSE + SVM (%)		LSE + LR (%)		SVM Jaccard (%)	
EstateFP	**100**	**1.7**	**100**	**1.9**	**90**	**0.7**
ExtFP	40	−0.0	**40**	**0.0**	20	−0.4
KlekFP	10	−0.7	10	−0.8	**40**	**0.0**
KlekFPCount	**70**	**0.7**	**80**	**0.9**	**100**	**1.3**
MACCSFP	**100**	**0.7**	**100**	**0.7**	**90**	**0.7**
PubchemFP	0	−1.0	20	−0.8	20	−0.0
SubFP	**100**	**2.5**	**90**	**2.5**	**90**	**1.1**
SubFPCount	**100**	**3.0**	**100**	**3.1**	**100**	**2.2**
Best	40	−0.4	50	−0.3	**60**	**0.0**

Joint BAC results is reported in 3, where each row corresponds to one of the 10 evaluated proteins, and each column corresponds to one of the 8 fingerprints. The last column represents maximum over all representations of a given compound.

It is important to note that LSE + SVM and LSE + LR are in the **vast majority** of cases better than much more complex SVM RBF model (80% of experiments, with at most 0.7% average BAC lost). This is a strong empirical argument that we can use simple, robust LSE + LR model instead in such tasks. Second, there is no significant difference between LSE + SVM and LSE + LR, which is interesting as Logistic Regression is again much simpler model. It might be the result of very dense data representation (we recall that LSE maps input data to just 8 dimensions), while SVMs seems to work better than LR on sparse data with many redundant (highly correlated) features. Finally, kNN is surprisingly strong model given its simplicity and seems to be underestimated in the cheminformatics field [1].

Testing complexities of our LSE models are drastically lower. Models only need to remember separating hyperplane and bias. Other SVMs have memorised up to 92% of training data as support vectors (on average), which have to be used during classification as the kernelized SVM rule for labeling given point iterates over all support vectors.

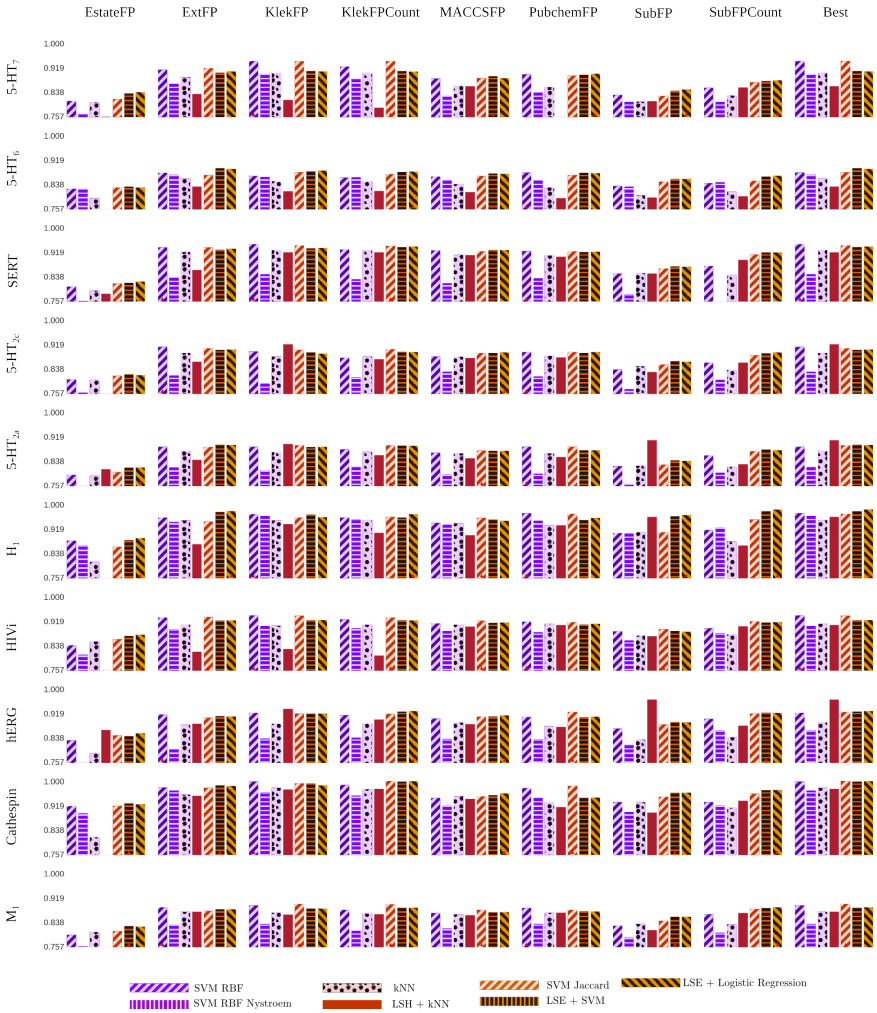


Figure 3. Results. The last two bars represent LSE + SVM and LSE + LR models.

Obviously single fact that models using local knowledge are stronger is not definitive proof of SVM RBF simplicity, or in general degenerative characteristics of data. Here we present additional justification of our claim, which together with classifiers accuracies comparision is in our view strong enough evidence to support our claim. First, number of support vectors memorized by SVM RBF and SVM RBF Nystroem is big, on average SVM RBF memorizes as support vectors 90% of training data, and

SVM RBF Nystroem 92%. This means that SVM RBF is learning something very heavily dependent on local neighbourhood, which explains why often kNN results are similar. Second, on average SVM RBF evaluated on incorrectly classified compounds by LSE + SVM achieves 34.7% BAC, which is worse than random classifier. This indicates that SVM RBF is learning similar concepts to classifier using only local knowledge.

Additional result of our research is coming up with a robust, order of magnitude faster model, that additionally allows for online learning. Table 2 presents training time of particular models and model parameters estimation as the average number of training points being used during classification.

Training of Jaccard SVM is obviously quadratic as it requires kernel computation. Computation of LSE is approximately $O(N^{1+p})$, but for such a small dataset it is comparable. It is also due to the fact that kernel computation is done via efficient C++ code, and our LSH is prototypical and done largely in much slower Python. Training Jaccard SVM took on average $5660s$ per experiment, whereas LSE computation took $2630s$. However, on the largest protein LSE computation took $6000s$ and kernel computation took 4 times longer (7 hours).

It is very interesting to note that LSE + LR has almost identical accuracy as LSE + SVM and also has a very low training time (which is almost equal to the time of construction of the LSE embedding). All of the results are very promising, because main time complexity comes from constructing embedding which can be significantly improved.

Table 2. Model complexities as measured by the number of parameters used during classification of the new point. d is fingerprint size and h is Nystroem feature space size (in our experiments set to 100).

Model	Model parameters	Training time [h]	Testing time [h]
SVM RBF	$\sim 2000 \cdot d$	233.27	21.8
SVM RBF Nystroem	$\sim 2000 \cdot h$	67.10	2.5
SVM Jaccard	$\sim 1000 \cdot 8$	11.69	0.4
LSE + SVM	$\sim 100 \cdot 8$	27.60	0.0
LSE + LR	$\sim 100 \cdot 8$	16.1	0.0

4. Conclusions

The main purpose of this paper was to address the following question *Is Support Vector Machine with RBF kernel learning any complex data patterns exploiting compound activity or does it degenerate to nearly nearest neighbour search?*. We constructed method which selects models from the family of hypotheses limited to the nearest neighbour search with Jaccard similarity and showed that even though SVM RBF

is capable of returning models from outside of this set, for the particular problem of drugs activity prediction with use of fingerprints representation **it does not**. The most probable reason of this phenomen is a strong violation of the i.i.d. assumption during dataset generation. In other words, samples are generated using a complex pattern emerging from pharmacologists' publication practices and this creates points distribution that cannot be well modeled using classical machine learning methods. We have shown nearly equivalent model in terms of both achieved results and represented knowledge and the answer is (assuming representativness of used proteins and fingerprints):

> Support Vector Machine with RBF kernel degenerates to nearly nearest neighbour search when applied to compound activity prediction.

As an additional result we proposed a classifier consisting of embedding step and linear model. Low-dimensional embedding technique maps fingerprints to 8 real dimensions resulting in a simple, efficient representation. A linear model (in particular Logistic Regression) using this representation behaves as well as state of the art method, while in the same time is much simpler and conveniently allows for fast online training.

Acknowledgements

The paper was partially funded by National Science Centre Poland Found grant no. 2013/09/N/ST6/03015.

5. References

[1] Kurczab R., Smusz S., Bojarski A.J., *Evaluation of different machine learning methods for ligand-based virtual screening.* J. Cheminformatics, 2011, 3(S-1), pp. P41.

[2] Cortes C., Vapnik V., *Support-vector networks.* Machine Learning, 1995, 20(3), pp. 273–297.

[3] Vapnik V., *The nature of statistical learning theory.* Springer, New York, 2000.

[4] Berlinet A., Thomas-Agnan C., *Reproducing kernel Hilbert spaces in probability and statistics.* Vol. 3. Springer, London, 2004.

[5] Drineas P., Mahoney M.W., *On the nyström method for approximating a gram matrix for improved kernel-based learning.* Journal of Machine Learning Research, 2005, 6, pp. 2153–2175.

[6] Joachims T., Training linear svms in linear time. In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2006, pp. 217–226.

[7] Rajaraman A., Ullman J.D., *Mining of massive datasets.* Cambridge University Press, Cambridge, 2011.

[8] Lewis D.D., Gale W.A., A sequential algorithm for training text classifiers. In: *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, Springer-Verlag New York, Inc., 1994, pp. 3–12.

[9] Swamidass S.J., Chen J., Bruand J., Phung P., Ralaivola L., Baldi P., *Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity.* Bioinformatics, 2005, 21(suppl. 1), pp. i359–i368.

[10] Yap C.W., *Padel-descriptor: An open source software to calculate molecular descriptors and fingerprints.* Journal of Computational Chemistry, 2011, 32(7), pp. 1466–1474.

[11] Smusz S., Czarnecki W.M., Warszycki D., Bojarski A.J., *Exploiting uncertainty measures in compounds activity prediction using support vector machines.* Bioorganic & Medicinal Chemistry Letters, 2015, 25(1), pp. 100–105.