Computational Intelligence, Volume 59, Number 000, 2010

Detonation Classification from Acoustic Signature with the Restricted Boltzmann Machine

YOSHUA BENGIO
NICOLAS CHAPADOS
OLIVIER DELALLEAU
Université de Montréal, Dept. of Computer Science and Operations Research, QC, Canada

HUGO LAROCHELLE
University of Toronto, Dept. of Computer Science, ON, Canada

XAVIER SAINT-MLEUX
CHRISTIAN HUDON
ApSTAT Technologies, Montreal, QC, Canada

JÉRÔME LOURADOUR
A2iA SA, Paris, France

We compare the recently proposed Discriminative Restricted Boltzmann Machine to the classical Support Vector Machine on a challenging classification task consisting in identifying weapon classes from audio signals. The three weapon classes considered in this work (mortar, rocket and rocket-propelled grenade), are difficult to reliably classify with standard techniques since they tend to have similar acoustic signatures. In addition, specificities of the data available in this study makes it challenging to rigorously compare classifiers, and we address methodological issues arising from this situation. Experiments show good classification accuracy that could make these techniques suitable for fielding on autonomous devices. Discriminative Restricted Boltzmann Machines appear to yield better accuracy than Support Vector Machines, and are less sensitive to the choice of signal preprocessing and model hyperparameters. This last property is especially appealing in such a task where the lack of data makes model validation difficult.

Key words: Discriminative Restricted Boltzmann Machine, Support Vector Machine, Detonation classification

1. INTRODUCTION

The Discriminative Restricted Boltzmann Machine (DRBM) has been recently proposed as a state-of-the-art statistical machine learning tool in supervised and semi-supervised classification settings (Larochelle and Bengio, 2008). Until now, empirical evaluation of this algorithm has been mostly limited to image or text data (Larochelle and Bengio, 2008; Marin-Jimenez *et al.*, 2009). In this paper, we compare it with the popular Support Vector Machine (SVM, see Cortes and Vapnik, 1995) on a challenging classification task consisting in performing automatic target classification of impulsive sources such as detonations of different kinds of weapons. This task is difficult since a large number of factors affect the propagation of the detonation signal from the impulsive source to the microphone. In particular, we note: (1) the distance between receiver and source, (2) the presence of obstacles on the terrain and nature of the ground, (3) the amplitude of the source, (4) the time of day, and (5) the meteorological conditions (cloud cover, wind, and humidity).

In addition to the acoustic signature of the weapon at the source, the factor believed to be most determinant in the recorded signal is the nature of the ground along the trajectory from the explosion to the microphone. Because there are many possible combinations of these factors, an ideal data collection would involve as many different combinations of these as possible, and in particular as many different trajectories and ground types as possible. Unfortunately, the data currently at our disposal for this study comes from only three different proving grounds with samples recorded during a small number of sessions, as will be detailed in Section 2.1. This makes this task challenging in two different ways:

- statistical models are prone to overfitting on specific characteristics of the data samples, that would not generalize to new recording locations and atmospheric conditions;
- it makes the evaluation of these models' reliability considerably more difficult, because of the lack of data samples recorded in truly different conditions (which would be required to be able to estimate generalization ability).

Address correspondence to Yoshua Bengio, DIRO, Université de Montréal, P.O. Box 6128, Centre-Ville Branch, Montreal, QC, H3C 3J7, Canada.

FIGURE 1. Examples of some audio samples with high signal-to-noise ratio from the three weapon classes.

One of our contributions in this paper regards statistical and methodological issues related to this problem of data scarcity: how do we fairly evaluate performance? How do we compare different classifiers in statistically reliable ways? Here, we propose a carefully-designed experimental setting in order to properly evaluate the generalization performance of classifiers as if they were deployed on the field. We also analyze our experimental results by systematically assessing their statistical significance.

Discriminant classifiers, such as SVMs, focus on learning the classification decision. In contrast, *generative classifiers*, like the (non-discriminative) Restricted Boltzmann Machine (RBM), try to capture the actual distribution of signals. Potential advantages of generative classifiers include the ability to take advantage of unlabeled data, i.e. signals for which the weapon class is unknown. They also have the potential to learn on their own a more refined representation of the input signal, one that captures the factors that explain the variations in the data. They have been proposed recently as components of so-called *deep architectures* (Hinton *et al.*, 2006; Bengio *et al.*, 2007; Ranzato *et al.*, 2007; Bengio, 2009), an approach that departs from existing non-parametric learning algorithms in order to learn the kind of highly-varying functions that are presumably necessary in artificial intelligence tasks. The DRBM aims to take advantage of both approaches, by combining the generative criterion of the RBM with a discriminative one, in order to achieve superior classification accuracy.

The purpose of this paper is to compare SVMs and DRBMs on the classification task of discriminating between three main types of weapons studied here: `ROCKET`, `RPG`[1] and `MORTAR` explosions. Previous work on the same kind of data has mostly been focused on automatic detection of weapon firing and source localization (see e.g. Bedard and Pare, 2003; Naz and Marty, 2006), while for classification tasks similar to the one we are interested in, researchers have been using neural networks (Desai *et al.*, 2006, 2007; Morcos *et al.*, 2008), as well as Gaussian mixtures and Maximum Entropy models (Smith, 2006). Our experiments explore interactions of classifier choice (SVM or DRBM) with various segmentation and preprocessing choices. We find that although both SVMs and DRBMs can achieve comparable *best* performance, DRBMs perform slightly better overall, and have the significant advantage of being less sensitive to the choice of preprocessing and algorithm hyperparameters. These results thus demonstrate the usefulness of the DRBM in such a setting where model evaluation is difficult due to the specificities of the data.

This paper is organized as follows: in Section 2 we provide an overview of the data and signal processing techniques employed; followed in Section 3 by the methodological issues we face for a fair comparison between classification algorithms; in Section 4 we review the formulation of the SVM and DRBM algorithms employed; statistical methods we use for the purpose of model evaluation and comparison are described in Section 5, and experimental results follow in Section 6. Finally, Section 7 concludes and suggests avenues for further research.

2. DATA OVERVIEW AND PREPROCESSING

2.1. Overview of Available Data

The acoustic data and associated meta-data that we have used to train and test statistical models come from several different exercises that took place in 2004 and 2005. All of the data that we have used was recorded by ground-based tetrahedral microphone arrays, and includes recordings for different types of weapons, recorded at separate locations and at different dates and times; each single detonation event may have been recorded by several different arrays. The sampling rate of each of the four microphones on an array is 1001.6 Hz. Examples of audio samples recorded by a single microphone are shown in Figure 1. From the point of view of the learning algorithm, the four signals representing the same detonation recorded by a single array are seen as four different data samples. The model's prediction for a recorded detonation is thus taken as the average of the individual model's predictions on these four samples (which are always kept together when splitting the dataset into train and test subsets).

From the various data sources, we have built a dataset containing the acoustic signatures of all unique launch events for weapons of types `MORTAR`, `ROCKET` and `RPG`. After a manual screening of the data to remove the lowest-quality

[1] Rocket-propelled grenade.

samples, this dataset contains a total of 636 different launch signatures, each having four individual signals (from the four microphones of a tetrahedral array). The data is not publicly available for download, but may be obtained by contacting the authors.

There is a severe class imbalance between the three main weapon types: `ROCKET`s account for 5.8% of all signatures, `RPG`s account for 4.7% and `MORTAR`s for the remaining 89.5%, which means that there are at least 15 times more signatures available for `MORTAR`s than for any other class. This is a serious issue with multiple implications which are discussed throughout this paper. Such a severe imbalance in the available data is not only present for the target classes on which we wish to do classification, but also for the times and locations of the events recorded. As an example, 59% of the signatures come from the Yuma proving ground, but less than 1% come from the Dahlgren Naval Surface Warfare Center, and all the data from Dahlgren is for `RPG`s and was collected on two consecutive afternoons. Also, all `ROCKET` signatures available come from the same proving ground, Aberdeen, and these were also recorded on two consecutive days. Section 3.1 explains the different ways in which we have chosen to split this data in order to obtain a scenario which is as close as possible to what would happen in the field if a classification model were to be deployed.

When working on this dataset, one also needs to take into account the fact that some of the launch events are represented by more than one signature. This is because there is more than one tetrahedral array recording all of the events from an exercise, and therefore we get as many signatures as there are arrays for each launch event. This can be important when designing an experimental setup meant to evaluate the performance of a model on new explosions, a topic which is discussed in Section 3.1.

2.2. Signal Segmentation

As the experiments presented in Section 6 illustrate, the segmentation can have a significant impact on classifier performance. Two segmentations are available for the signals we consider. The first one, called `ARLTruncated`, is a fully manual segmentation giving both the start and end points of the detonation in an audio file, which is truncated accordingly before being processed to perform the classification task.

The second one, called `ApPeakNoTrunc`, is a semi-automatic segmentation providing only the start point of the detonation: an algorithm for automatic edge detection in the signal is run, in order to suggest where the detonation would start. Then a human either validates it as the main peak in the signal, or manually sets the main peak if the one being suggested is not considered appropriate (which happens mostly in noisy samples). The start point of the signal is then set to this main peak, while no end point is defined (the signal is not truncated afterwards). The algorithm we use for automatic edge detection is a discrete-time formulation of the continous-time method proposed by Engelberg (2008), based on the idea of emphasizing high frequencies in the Fourier domain so as to detect points where the signal value changes in an (approximately) discontinuous way.

2.3. Preprocessing by Spectral Features

We ran preliminary experiments with various preprocessing techniques in order to decide which one to use in our comparisons between the baseline SVM and DRBM. In order to ensure fairness w.r.t. the SVM, the preprocessing technique was chosen to be the one giving the best results with this learning algorithm (which will be described in more details in Section 4.1). We compared the Fast Fourier Transform (FFT), the Haar and Symlet 6 wavelets (see e.g. Graps, 1995), cepstral coefficients (MFCC, which are commonly used in speech and signal processing applications, see e.g. Bogert *et al.*, 1963; Childers *et al.*, 1977), and Shift-Invariant Sparse Coding (SISC), a sparse feature extraction algorithm that is robust to the translation of signals in time (Grosse *et al.*, 2007). For the sake of simplicity we only give specific details about the FFT preprocessing since, as shown in Figure 2, it is the one yielding the best performance,[2] and is thus what we use in the reminder of this paper.

The spectral features obtained from the Fast Fourier Transform are computed as follows. We extract 35 spectral coefficients on frames starting according to the chosen segmentation. Frame length varies between 256 and 1024 samples, given a sampling rate of 1001.6 Hz, and Hamming windowing is applied. Spectral coefficients are computed by applying a bank of triangular filters on the result of the FFT. The filters are normalized in power and spread log-uniformly between 0 and 450 Hz, a scale that focuses more on low frequencies than the mel scale commonly used in speech recognition with cepstral coefficients (Deller *et al.*, 1999; Quatieri, 2001). This is because initial experiments showed that most of the relevant spectral information to discriminate between detonation signatures lies in frequencies lower than the ones often used in speech processing. As a result, the log-scale preserves more of the important features for discrimination than the mel scale, the latter being quasi-linear over the 0–500 Hz band.

We vary the number of frames between one and four, depending on the experiments; the delay between frames is kept constant at 128 samples (a value that was found to consistently give among the best results on this task). When

[2]Note that in these preliminary experiments, the data was shuffled randomly, which does not result in a reliable estimate of generalization error, as explained in Section 3.1 and illustrated by Figure 3. This is why the best accuracies are significantly better than those reported later in experiments.

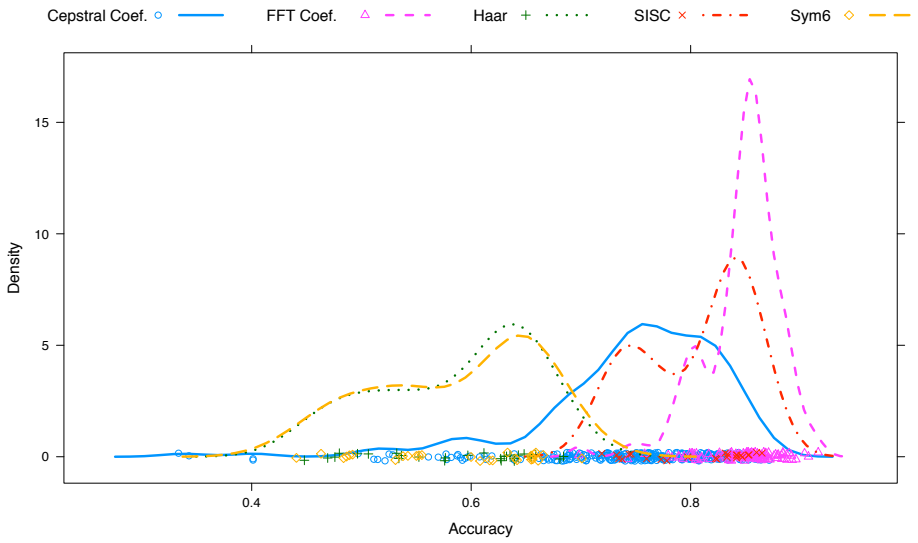

FIGURE 2. Distribution of classification accuracies for various preprocessing types using a baseline SVM classifier. For each type of preprocessing, relevant hyper-parameters are varied leading to various data points which are used to build a kernel density estimate, plotted above. Overall, the FFT preprocessing exhibits the highest classification accuracy.

using more than one frame, first derivatives are added to the features and are approximated by taking the difference between the spectral coefficients of two consecutive frames.

In some contexts, we find it helpful to take the logarithm of the spectral coefficients before building the final feature vector (in this case, first derivatives are computed on the log-coefficients as well). Section 6 presents results both with and without this variant.

3. METHODOLOGICAL ISSUES

3.1. Correctness Issues and Data Splits

A classifier is meant to be used in the field on new data which are not available at the time of developing the classifier and we call such data the *field data*. For a cross-validation procedure to provide a correct estimation of future classification performance, a basic assumption needs to be verified: the relation between the test data and the training data should be representative of the relation between field data and the training data. If this is not the case, there is a risk of not noticing overfitting on the data available to build and evaluate the model. Figure 3 illustrates this danger: it is possible to obtain around 95% *test* accuracy by randomly shuffling the signals between the training and test data splits. However, this is "cheating" because, for each detonation in the test set, there is almost always another detonation in the training set that was recorded in similar conditions (e.g. same day, same location of weapon and sensor array, or even same event recorded from a different array).

Because of this, we carried out performance evaluation within two different testing frameworks, defined as follows:

- REALISTIC-BY-DAY: in this experimental setup, we ensure that a group of explosions recorded during the same day is never split between the training and test sets. This is close to the expected use of such an automatic system, which will be trained from data collected on a few military proving grounds and tested on new samples recorded under very different conditions. In this setup, we perform five-fold cross-validation under the above constraint that data recorded during the same day must not be shared by both the training and test sets. In order to reduce the impact of randomness on the performance measure, the five-fold cross-validation is repeated five times, and we report the average performance over these five repetitions.
- REALISTIC-BY-RANGE: the above REALISTIC-BY-DAY setting would be sufficient to evaluate performance if conditions varied significantly from one day to another. Unfortunately, this is not always the case: for instance, it can happen that recording sessions taking place over two days or more use the same positioning of weapons and sensor arrays. As a result, explosions recorded during day one of the session may look very similar to those recorded during day two, leading to over-estimating the test accuracy. This motivated the definition of a second experimental setup, called REALISTIC-BY-RANGE, where the constraint is instead that a group of explosions that share the same positioning of weapon and sensor array is never split between the training and test sets. A five-time repetition of five-fold cross-validation is used to assess a model's accuracy, similar to the one described in the REALISTIC-BY-DAY setting.

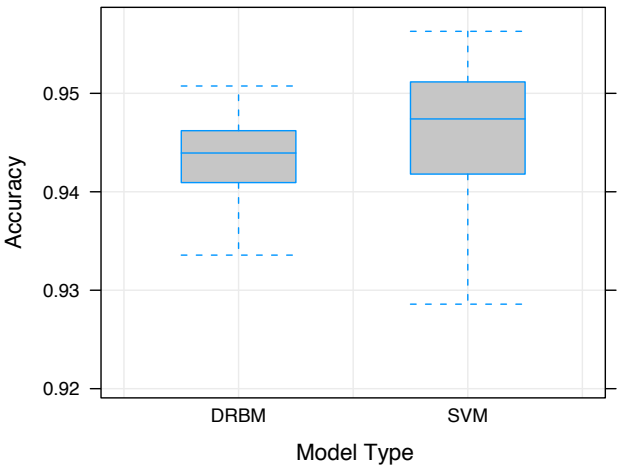

FIGURE 3. When the training and test data are split in a random way, very high test accuracies (around 95% correct) can be achieved by both the DRBM and SVM. However, these accuracies do not represent the kind of performance the model would achieve on new data recorded in different conditions.

One should keep in mind that the second setting (REALISTIC-BY-RANGE) is not ideal either. In particular, when the same explosion is recorded by two sensing arrays placed in different positions, these two recordings may be split among the training and test sets, which could make the identification task easier than what would be expected for field data. Due to the lack of available data, it is however impossible to design a basic cross-validation procedure that would ensure the training and test samples are totally independent. The two realistic settings described here thus are *approximations* that can give us a reasonable idea of a classifier's performance, but should not be trusted blindly. The differences between the performances in the two settings may also be used to better understand what a model is learning: for instance, if a classifier has good test accuracy in the REALISTIC-BY-DAY setting but not in REALISTIC-BY-RANGE, it may be because it is basing its prediction on the relative position of weapons and sensors (which may not always vary from one day to another), rather than on the acoustic features themselves.

3.2. Hyperparameter Selection

When a number of models are trained with different hyperparameters, the issue of *model selection* arises: which set of hyperparameters should be selected as being the "best-performing" one? Moreover, in addition to selecting hyperparameters, we wish to estimate the *expected future performance of a model*. Unfortunately, the best accuracy on the validation set (best across different values of the hyperparameters tried) can be optimistically biased, because of the selection across different hyperparameter values.

To avoid that bias, a rigorous procedure would rely on *two-level cross-validation*. Unfortunately, given the very limited amount of data in some of the classes that we must classify (as explained in Section 2.1), we had to rely on a different procedure since the two-level cross-validation may result in severe instability: due to sampling variability or splitting constraints[3], instances from the minority classes can be nearly (or completely) absent from some internal validation folds, resulting in a disproportionate influence of those cases on hyperparameter choice. This makes hyperparameter selection by two-level cross-validation very fragile and exhibiting a large variance.

As an alternative, we rely on the following systematic procedure drawing from classical inference to select sets of "comparably well-performing" hyperparameters for the purpose of model comparison:

(1) On test results arising from a one-level cross-validation (performed in one of the two settings described in Section 3.1), we carry out an ANOVA to determine statistically-significant main effects and interactions (the ANOVA method is detailed in Section 5.1).
(2) We keep all combinations of hyperparameters that are not statistically significantly different from the best-performing set at the 5% level. This set of hyperparameters yields an empirical distribution of test accuracies.
(3) Comparisons between models are performed by contrasting aspects of their respective accuracy distribution, for instance its mean, median or variance.

[3]Constraints on which data samples are allowed to be split between the train and test parts, as defined in the REALISTIC-BY-DAY and REALISTIC-BY-RANGE settings introduced previously.

4. CLASSIFICATION ALGORITHMS

We experimented with two learning algorithms for classification: the Support Vector Machine and the Discriminative Restricted Boltzmann Machine. While the former is widely used and has demonstrated great performance for a wide range of problems, the latter is more recent but has been shown to be more appropriate for certain problems with high-dimensional inputs, such as image and text data.

4.1. Support Vector Machines

The Support Vector Machine (SVM) is a well-known classification algorithm (Cortes and Vapnik, 1995; Vapnik, 1998), and serves as a baseline in our experiments: it has shown great success in solving classification problems in the last decade. The typical binary SVM classifier takes a ± 1 decision based on the sign of a discriminative function f subject to some constraints. The quantity minimized during the training of a binary SVM classifier is:

$$\min_{\text{constrained } f} \quad \underbrace{\mathcal{C} \sum_{\mathbf{z}^{(i)} \in \mathcal{T}} \left(1 - y^{(i)} f(\mathbf{z}^{(i)})\right)_+}_{hinge\ loss\ (bias)} \quad + \quad \underbrace{\| f \|}_{margin\ (variance)} \tag{1}$$

where $\mathbf{z}^{(i)}$ is a real-valued input sample, $y^{(i)}$ its associated target (± 1), and $\mathcal{C}$ a regularization parameter which controls the bias-variance tradeoff. The margin is written as the norm of f. Another important aspect of the SVM is the possibility to use any kernel, not only linear ones, to be able to build complex model. If we consider the family of discriminative functions f that belong to the Reproducing Kernel Hilbert Space generated by a positive semi-definite kernel k, the representer theorem (Kimeldorf and Wahba, 1971) lets us write the solutions of optimization problem (1) in the form:

$$f(\mathbf{z}) = \sum_{\mathbf{z}^{(i)} \in \mathcal{T}} \mathbf{a}_i \, y^{(i)} \, k(\mathbf{z}^{(i)}, \mathbf{z}) + b \tag{2}$$

where $\mathbf{a}_i$ is a positive coefficient, and b a real number. Maximizing the margin tends to force some $\mathbf{a}_i$ to be zero. The optimization problem (1) can also be written in the dual form:

$$\begin{aligned} \min_{\mathbf{a}_i} \quad & \sum_{i=1}^{N_\mathcal{T}} \sum_{j=1}^{N_\mathcal{T}} \mathbf{a}_i \mathbf{a}_j \, y^{(i)} y^{(j)} \, k(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}) - \sum_{i=1}^{N_\mathcal{T}} \mathbf{a}_i \\ \text{subject to} \quad & 0 < \mathbf{a}_i < \mathcal{C} \end{aligned} \tag{3}$$

To train an SVM classifier that solves this optimization problem we used libSVM (Chang and Lin, 2001), which is based on an efficient Sequential Minimal Optimization (SMO) method as formulated by Fan *et al.* (2005).

4.1.1. *Kernel.* The choice of the kernel is as crucial as the choice of the input representation, and both are tightly related. In all our experiments, inputs are fixed-size vectors that summarize an audio signal.[4] We then considered in our experiments three widely used families of vector kernels:

- The *linear* kernel

$$k(\mathbf{z}^{(1)}, \mathbf{z}^{(2)}) = {\mathbf{z}^{(1)}}' \mathbf{z}^{(2)} \tag{4}$$

- The *polynomial* kernel

$$k(\mathbf{z}^{(1)}, \mathbf{z}^{(2)}) = (r + {\mathbf{z}^{(1)}}' \mathbf{z}^{(2)})^p \tag{5}$$

- The *RBF* kernel, which is a Radial Basis Function

$$k(\mathbf{z}^{(1)}, \mathbf{z}^{(2)}) = e^{-\gamma \|\mathbf{z}^{(1)} - \mathbf{z}^{(2)}\|^2} \tag{6}$$

Normalizing the inputs is also important to have good and stable results, and the normalization process can be seen as part of the kernel design. We found in our experiments that the best normalization usually consists in shifting and rescaling each input component so that it fills the range $[-1, 1]$ over the training set, and this is the normalization we use throughout this paper.

4.1.2. *Hyper-optimization.* The kernel hyperparameters (r, p in (5) and γ in (6)), as well as the regularization hyperparameter $\mathcal{C}$ in (1) and (3), are *automatically* chosen based on an "internal" 3-fold cross-validation on the training set. The automatic search is based on heuristics that have been found to work well in practice; the main lines are summed up in the following algorithm.

[4] In the context of detonation classification, the choice of a fixed-size input vector—particularly in conjunction with spectral features—is appropriate since there is low variance in the duration of detonation events, and here we can rely on accurate segmentation of these events.

(1) Choose 3 initial values for each hyperparameter and consider all possible combinations of these values. For instance, for the *RBF* kernel, $\mathcal{C} \in \{0.1, 1, 10\}$ and $\gamma \in \{\gamma_0/10,\ \gamma_0,\ 10\gamma_0\}$, where $\gamma_0 = \frac{1}{(2\,d\,\sigma^2)}$ is the value recommended by Schölkopf *et al.* (1999) , where d denotes the input dimensionality, and σ the average standard deviation for all inputs.
(2) Compute the cross-validation performance for each new candidate set of hyperparameters.
(3) If, for at least one hyperparameter, the best performance is obtained for a minimal or maximal value p of this hyperparameter amongst the ones tried with the same optimal values for other hyperparameters, then choose two values "around" p, and go to step 2. For example, consider the case of optimizing γ for an SVM classifier using the *RBF* kernel. Assume that the optimal value of γ is γ_1 and the values already tried are $\gamma_1 < \gamma_2 < \gamma_3 < \ldots$ Then the next candidate values will be the geometric mean $\sqrt{\gamma_1\,\gamma_2}$ and the smaller value $\frac{\gamma_1}{\gamma_2}\gamma_1$.
(4) Stop when a locally optimal set of hyperparameters has been found and retrain the SVM classifier on the whole training set with these hyperparameters.

Also, an important aspect of the hyper-optimization is the method used for generating folds during the internal 3-fold cross-validation. The better the match between fold generation in the cross-validation and the final evaluation, the higher the chances of having good results on the final evaluation. For example, if the evaluation is done with the REALISTIC-BY-DAY setting (see 3.1), then the folds are determined so as not to split a group of recordings from the same day in different folds.[5] Experiments showed that this strategy improves performance with respect to random folds.

4.1.3. *Multiclass SVM.* In their basic form, SVMs are fundamentally binary classifiers, so several methods have been proposed to generalize them to more than two classes. Assume there are $C > 2$ classes, such that labels are in $\{1, \ldots, C\}$. The two main options are (Hsu and Lin, 2002):

- *one-against-all* scheme: for each class $i \in \{1, \ldots, C\}$, one SVM classifier $f_{i,*}(\mathbf{z})$ is trained to discriminate between this class and all the others.
- *one-against-one* scheme: one SVM classifier $f_{i,j}(\mathbf{z})$ is trained for each pair of classes $\{i, j\}$, $i \neq j$, with i being considered the positive class (+1 label), and j the negative one (-1 label).

In both cases, outputs have to be combined to decide the dominant class (the one with the highest output).

In our experiments, both strategies performed almost the same, provided that the combination of SVM output is adapted to the strategy. The *one-against-all* scheme works well when the output for each class is simply $f_{i,*}(\mathbf{z})$. The *one-against-one* scheme works well when the output for each class i is $\sum_{j \neq i | f_{i,j}(\mathbf{z}) > 0} \mathrm{sigm}(f_{i,j}(\mathbf{z}))$, where sigm is the logistic sigmoid

$$\mathrm{sigm}(x) = \frac{1}{1 + e^{-x}}.$$

4.2. Discriminative Restricted Boltzmann Machines

We now turn to the second classifier that we considered. We review the theoretical formulation of DRBMs, building from the simpler Restricted Boltzmann Machine (RBM) model.

An RBM is an undirected generative (probabilistic) model that uses a layer of hidden variables to model a distribution over visible variables. Such models are most often trained to only model the inputs of a classification task, so as to extract features from the data in an unsupervised fashion (see e.g. recent work on audio classification by Lee *et al.*, 2009). But they can also model the joint distribution of the inputs and associated target classes (e.g. as done by Larochelle and Bengio (2008) and Tieleman (2008), or in the last layer of a deep neural network as shown by Hinton *et al.* (2006)). In this paper, we present experiments with such a joint model, which is depicted in Figure 4.

An RBM with H hidden units is a parametric model of the joint distribution between a layer of hidden variables (often referred to as features) $\mathbf{h} = (\mathbf{h}_1, \ldots, \mathbf{h}_n)$ and the visible variables made of the inputs $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_d)$ and the target y, that takes the form

$$p(y, \mathbf{z}, \mathbf{h}) \quad \propto \quad e^{-E(y, \mathbf{z}, \mathbf{h})}$$

where

$$E(y, \mathbf{z}, \mathbf{h}) \quad = \quad -\mathbf{h}'\mathbf{W}\mathbf{z} - \mathbf{b}'\mathbf{z} - \mathbf{c}'\mathbf{h} - \mathbf{d}'\vec{y} - \mathbf{h}'\mathbf{U}\vec{y} \tag{7}$$

with parameters $\Theta = (\mathbf{W}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{U})$ and $\vec{y} = (1_{y=i})_{i=1}^{C}$ for C classes. Though RBMs are usually applied on problems with binary inputs, they can easily be generalized to real-valued inputs (Welling *et al.*, 2005). We discuss how such a modification applies to our experiments later in this section.

[5] This is not always possible due to the scarcity of data, and for some folds we had to allow this constraint to be violated in the internal cross-validation.

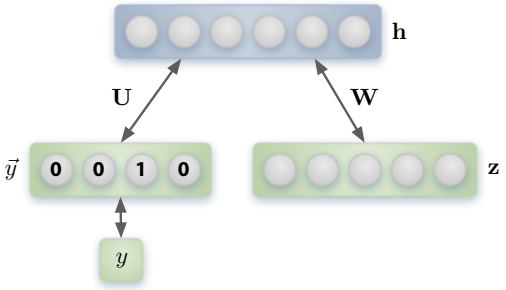

FIGURE 4. Restricted Boltzmann Machine modeling the joint distribution of inputs $\mathbf{z}$ and target class y (represented as a one-hot vector by $\vec{y}$). The current state of the hidden units is labeled by $\mathbf{h}$.

Consider for now that the input variables $\mathbf{z}$ are binary. It is simple to show that in an RBM, the conditional distributions between layers are as follows:

$$\begin{aligned} p(\mathbf{z}|\mathbf{h}) &= \prod_i p(\mathbf{z}_i|\mathbf{h}) \\ p(\mathbf{z}_i = 1|\mathbf{h}) &= \text{sigm}\left(\mathbf{b}_i + \sum_j \mathbf{W}_{ji}\mathbf{h}_j\right) \quad (8) \\ p(y|\mathbf{h}) &= \frac{e^{\mathbf{d}_y + \sum_j \mathbf{U}_{jy}\mathbf{h}_j}}{\sum_{y^*} e^{\mathbf{d}_{y^*} + \sum_j \mathbf{U}_{jy^*}\mathbf{h}_j}}. \quad (9) \end{aligned}$$

Equations (8) and (9) illustrate that the hidden units are meant to capture predictive information about the input vector as well as the target class. The conditional distribution of hidden units given inputs and target class, $p(\mathbf{h}|y, \mathbf{z})$, also has a similar form:

$$\begin{aligned} p(\mathbf{h}|y, \mathbf{z}) &= \prod_j p(\mathbf{h}_j|y, \mathbf{z}) \\ p(\mathbf{h}_j = 1|y, \mathbf{z}) &= \text{sigm}\left(\mathbf{c}_j + \mathbf{U}_{jy} + \sum_i \mathbf{W}_{ji}\mathbf{z}_i\right). \quad (10) \end{aligned}$$

Since an RBM defines a distribution over all of its variables, there is more than one strategy that can be used to train it. The most common one is known as *generative training*. Given a training set $\mathcal{T} = \{(\mathbf{z}^{(i)}, y^{(i)})\}$ of $N_\mathcal{T}$ pairs of input feature vectors and targets, we can train this generative model by considering the minimization of the negative log-likelihood of that data:

$$\mathcal{L}_{gen} = -\sum_{i=1}^{N_\mathcal{T}} \log p(y^{(i)}, \mathbf{z}^{(i)}). \quad (11)$$

In order to minimize the negative log-likelihood (11), we would like an estimator of its gradient with respect to the model parameters. The exact gradient of a likelihood term, for any parameter $\theta \in \Theta$ can be written as follows:

$$\begin{aligned} \frac{\partial \log p(y^{(i)}, \mathbf{z}^{(i)})}{\partial \theta} &= -\mathbb{E}_{\mathbf{h}|y^{(i)},\mathbf{z}^{(i)}}\left[\frac{\partial}{\partial \theta} E(y^{(i)}, \mathbf{z}^{(i)}, \mathbf{h})\right] \\ &\quad + \mathbb{E}_{y,\mathbf{z},\mathbf{h}}\left[\frac{\partial}{\partial \theta} E(y, \mathbf{z}, \mathbf{h})\right]. \end{aligned}$$

Though the first expectation is tractable, the second one is not. Fortunately, there exists a good stochastic approximation of this gradient, called the contrastive divergence update (Hinton, 2002). This approximation replaces the second expectation by a sample generated after a limited number of Gibbs sampling iterations, with the sampler's initial state for the visible variables initialized at the training sample $(y^{(i)}, \mathbf{z}^{(i)})$. Even when using only one Gibbs sampling iteration, contrastive divergence has been shown to produce only a small bias for a large speed-up in training time (Carreira-Perpiñán and Hinton, 2005). Online training of an RBM thus consists in cycling through the training examples and updating the RBM's parameters according to Algorithm 1, where the learning rate is controlled by λ.

Computing $p(y, \mathbf{z})$ is intractable, but it is possible to compute $p(y|\mathbf{z})$, sample from it, or choose the most probable class. As shown by Salakhutdinov *et al.* (2007), for reasonable numbers of classes C (over which we must sum), this

Algorithm 1 Training update for RBM over $(y^{(i)}, \mathbf{z}^{(i)})$ using Contrastive Divergence.

Input: training pair $(y^{(i)}, \mathbf{z}^{(i)})$ and learning rate λ
{ Notation: $a \leftarrow b$ means a is set to value b
$a \sim p$ means a is sampled from p}

{Positive phase}
$y^0 \leftarrow y^{(i)}, \mathbf{z}^0 \leftarrow \mathbf{z}^{(i)}, \widehat{\mathbf{h}}^0 \leftarrow \mathrm{sigm}(\mathbf{c} + \mathbf{W}\mathbf{z}^0 + \mathbf{U}\vec{y^0})$

{Negative phase}
$\mathbf{h}^0 \sim p(\mathbf{h}|y^0, \mathbf{z}^0), y^1 \sim p(y|\mathbf{h}^0), \mathbf{z}^1 \sim p(\mathbf{z}|\mathbf{h}^0)$
$\widehat{\mathbf{h}}^1 \leftarrow \mathrm{sigm}(\mathbf{c} + \mathbf{W}\mathbf{z}^1 + \mathbf{U}\vec{y^1})$

{Update}
for $\theta \in \Theta$ **do**
$\theta \leftarrow \theta - \lambda \left(\frac{\partial}{\partial \theta} E(y^0, \mathbf{z}^0, \widehat{\mathbf{h}}^0) - \frac{\partial}{\partial \theta} E(y^1, \mathbf{z}^1, \widehat{\mathbf{h}}^1) \right)$
end for

conditional distribution can be computed exactly and efficiently, by writing it as follows:

$$p(y|\mathbf{z}) \quad = \quad \frac{e^{\mathbf{d}_y} \prod_{j=1}^{H} \left(1 + e^{\mathbf{c}_j + \mathbf{U}_{j,y} + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i}\right)}{\sum_{y^*} e^{\mathbf{d}_{y^*}} \prod_{j=1}^{H} \left(1 + e^{\mathbf{c}_j + \mathbf{U}_{j,y^*} + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i}\right)}. \tag{12}$$

Precomputing the terms $\mathbf{c}_j + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i$ and reusing them when at the time of computing the product

$$\prod_{j=1}^{H} \left(1 + e^{\mathbf{c}_j + \mathbf{U}_{j,y^*} + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i}\right)$$

for all classes y^* permits to compute this conditional distribution in time $O(Hd + HC)$.

However, in a classification setting, one is ultimately only interested in correct classification, not necessarily to have a good $p(\mathbf{z})$. Because the modeling assumptions for $p(\mathbf{z})$ implicitly made by the RBM can be inappropriate, it can then be advantageous to optimize directly $p(y|\mathbf{z})$ instead of $p(y, \mathbf{z})$ by considering the minimization of the following cost:

$$\mathcal{L}_{disc}(\mathcal{T}) = -\sum_{i=1}^{N_{\mathcal{T}}} \log p(y^{(i)}|\mathbf{z}^{(i)}). \tag{13}$$

This training strategy is called *discriminative training*, and we refer to RBMs trained according to $\mathcal{L}_{disc}$ as Discriminative RBMs (DRBMs).

A DRBM can be trained by contrastive divergence but since $p(y|\mathbf{z})$ can be computed exactly, we can compute the exact gradient:

$$\begin{aligned} \frac{\partial \log p(y^{(i)}|\mathbf{z}^{(i)})}{\partial \theta} \quad &= \quad \sum_j \mathrm{sigm}(o_{y^{(i)},j}(\mathbf{z}^{(i)})) \frac{\partial o_{y^{(i)},j}(\mathbf{z}^{(i)})}{\partial \theta} \\ &- \quad \sum_{j,y^*} \mathrm{sigm}(o_{y^*j}(\mathbf{z}^{(i)})) p(y^*|\mathbf{z}^{(i)}) \frac{\partial o_{y^*j}(\mathbf{z}^{(i)})}{\partial \theta} \end{aligned}$$

where $o_{y,j}(\mathbf{z}) = \mathbf{c}_j + \sum_k \mathbf{W}_{j,k}\mathbf{z}_k + \mathbf{U}_{j,y}$. This gradient can be computed efficiently and then used in a stochastic gradient descent optimization. This discriminative approach has been used previously for fine-tuning the top RBM of a Deep Belief Network (Hinton, 2007).

The advantage brought by discriminative training usually depends on the amount of available training data. Smaller training sets tend to favor generative learning and bigger ones tend to favor discriminative learning (Ng and Jordan, 2002). However, instead of solely relying on one or the other perspective, one can adopt a *hybrid discriminative/generative* approach simply by combining the respective training criteria. Though this method cannot be interpreted as a maximum likelihood approach for a particular generative model as in Lasserre *et al.* (2006), it proved useful here and elsewhere (Bouchard and Triggs, 2004). In this work, we used the following criterion (Larochelle and Bengio, 2008):

$$\mathcal{L}_{hybrid}(\mathcal{T}) = \mathcal{L}_{disc}(\mathcal{T}) + \alpha \mathcal{L}_{gen}(\mathcal{T}) \tag{14}$$

where the weight of the generative criterion is controlled by α. Here, the generative criterion can also be seen as a data-dependent regularizer for a DRBM. To train a DRBM in this context, we can use stochastic gradient descent and add for each example the gradient contribution due to $\mathcal{L}_{disc}$ with α times the stochastic gradient estimator associated with $\mathcal{L}_{gen}$ for that example.

For this paper, we used the DRBM with and without additional generative training. Also, since $\mathbf{z}$ is real valued,[6] we used *Gaussian visible units* (Welling *et al.*, 2005; Bengio *et al.*, 2007) for the inputs. Gaussian units are obtained by adding the quadratic term $\sum_{i=1}^{d} \mathbf{a}_i^2 \mathbf{z}_i^2$ in the energy function of Equation (7). From this new energy function, it can be shown that each conditional distribution $p(\mathbf{z}_i|\mathbf{h})$ is now a Gaussian distribution of mean $\boldsymbol{\mu}_i$ and variance parameter $\boldsymbol{\sigma}_i^2$:

$$\boldsymbol{\mu}_i = \frac{\mathbf{b}_i + \mathbf{W}'_{\cdot,i}\mathbf{h}}{2\mathbf{a}_i^2}, \qquad \boldsymbol{\sigma}_i^2 = \frac{1}{2\mathbf{a}_i^2}, \quad \forall i \in \{1, \ldots, d\}, \tag{15}$$

while the other conditional distributions of Equations (9), (10) and (12) remain the same.

In order to determine the number of iterations over the training set $\mathcal{T}$ to train our model, we first divide the original $\mathcal{T}$ into two parts $\mathcal{T}^{\frac{4}{5}}$ and $\mathcal{T}^{\frac{1}{5}}$ (according to a $\frac{4}{5}$ and $\frac{1}{5}$ split), where we train our model on $\mathcal{T}^{\frac{4}{5}}$ and use $\mathcal{T}^{\frac{1}{5}}$ to determine when to stop training using early-stopping (i.e. we stop when the accuracy on $\mathcal{T}^{\frac{1}{5}}$ starts dropping). Then, we look at the training conditional negative log-likelihood $\mathcal{L}_{disc}(\mathcal{T}^{\frac{4}{5}})$ that was reached, and retrain our model from scratch on the whole training set until either the same conditional negative log-likelihood is reached for $\mathcal{L}_{disc}(\mathcal{T})$, or we have performed a maximum number of iterations equal to twice as many iterations as what was required on $\mathcal{T}^{\frac{4}{5}}$.

5. STATISTICAL METHODS

This section contains a brief summary of some of the statistical methods that we repeatedly use throughout this paper, along with relevant references.

5.1. Analysis of Variance

The ANOVA is a statistical technique that is used to decompose the variability of a measured variable in terms of explanatory variables (*factors*), that can take different values (*levels*). In our case, we use ANOVAs to understand how the observed test performance of a classifier is affected by the levels of hyperparameters when those are varied within the experiment. The analysis of variance (and the related design of experiments) is a vast topic; see, e.g., Box *et al.* (2005) for more information.[7]

The fundamental purpose of an ANOVA is to test hypotheses about equality of means. We want to test the null hypothesis that the *mean performance* is equal for all levels of each factor (and possible interactions thereof). A rejection of the null hypothesis indicates that there are some levels for which performance is significantly different from others.[8] The ANOVA can consider the effect of each factor individually, as well as higher order interactions between two or more factors. In this paper, all ANOVA tables are presented along the following lines:

```
                           Df   Sum Sq  Mean Sq F value    Pr(>F)
Number of Windows           2 0.060123 0.030061 54.5114 < 2.2e-16 ***
Window Duration             2 0.010981 0.005491  9.9565 0.0001148 ***
Nb Windows : Duration Ix    4 0.002798 0.000699  1.2684 0.2875746
Residuals                  99 0.054595 0.000551
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In this example we report the results of an experiment with two factors: the number of FFT windows and the FFT window duration (see 2.3); we also test for interaction effects between these two factors (which we abbreviate by `Ix` in the table). The column `Df` in the ANOVA table stands for "degrees of freedom", and is one less than the number of levels in each factor (for main effects only). For example, the `Number of Windows` factor has three distinct levels, and thus two degrees of freedom. The `Residuals` degrees of freedom (last line) is computed as the total degrees of freedom (i.e. number of experimental results) minus the sum of the factor degrees of freedom: it is related to the observation variability that cannot be fit by the parameters of the ANOVA model. The `Sum Sq` column is the sum of squared deviations from the overall mean performance absorbed by the factor, and `Mean Sq` is `Sum Sq` divided by `Df`. For the ANOVA model to explain well the experimental results, we would like the `Mean Sq` associated with each experimental factor to be large relative to the residuals. This can be formally tested with a Fisher F-test. The column `F value` contains the values of the variance ratio statistics, namely the ratio of the `Mean Sq` of the factor being tested to the `Mean Sq` of the residuals. Finally, the column `Pr(>F)` is the p-value of observing such an extreme variance ratio, using Fisher's F distribution with the factor's degrees of freedom in the numerator, and the residuals' degrees of

[6] Inputs are normalized in the $[-1, 1]$ range.

[7] The excellent free online handbook edited by Croarkin and Tobias (2006) and maintained by the U.S. National Institute of Standards and Technology contains a wealth of information; section 7.4.3 explains ANOVAs in detail.

[8] But it does not tell us *which* levels are "better"; this necessitates the use of so-called *post hoc* analyses, such as the Tukey pairwise comparison procedure described next.

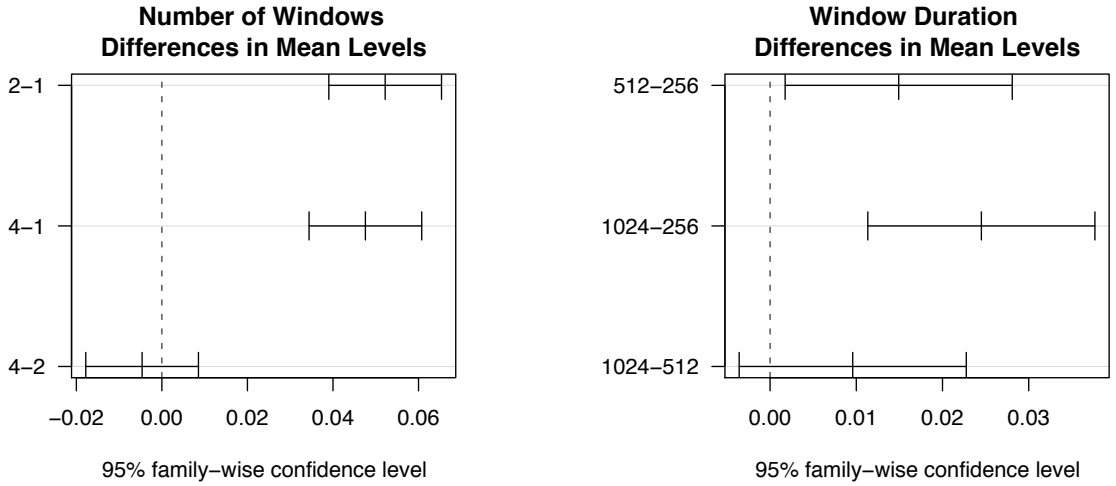

FIGURE 5. Graphical result of Tukey's **honestly significant differences** procedure. It presents the results of all pairwise mean differences between the levels of factors involved in an experiment (the levels being compared are shown on the left side), and associated confidence intervals on this difference. Here, it is significantly better to use either 2 or 4 windows (compared to a single window), with size either 512 or 1024 (compared to 256).

freedom in the denominator. A factor with a significant p-value indicates that it explains significantly more of the of the observation's variability than the residual noise level.

What this table tells us is the two main effects have a highly significant impact on performance (both their p-values are smaller than conventionally-accepted thresholds such as $p = 0.05$), but their interaction is not significant (since $p = 0.29 > 0.05$). Put differently, the performance can be approximated by the sum of the main effects, without a need to account for an additional term correcting for joint effect of a given number of windows and duration acting together.

Occasionally, higher-order interactions (greater than degree two) are necessary to represent complex empirical relationships between hyperparameters. In this paper, to avoid overloading the results, we include higher-order interactions in ANOVA tables only when they are statistically significant.

5.2. Tukey's Honestly Significant Differences

The ANOVA is helpful to understand which factors are driving performance; however, it only tells us that, e.g. the number of windows is significant, without telling us *which level* of a given factor produces the best performance (assuming no interactions). This is the purpose of secondary analyses, of which Tukey's method of *honestly significant differences* is used extensively in this paper.

Tukey's method of *honestly significant differences* (Hsu, 1996) is a way to perform all pairwise mean comparisons between the different levels of each factor in a single step, and control for experiment-wise error rate (also called the "family error rate"). Without going into details of mathematical procedure, the approach is very similar to repeated t-tests, except that it corrects for the occurrence of multiple comparisons.

In this paper, we present the results of Tukey's test in the form of plots of all pairwise mean differences between the levels of the factors involved in the experiment. An example corresponding the the previous ANOVA table appears in Fig. 5. From the figure, we note, for instance, that using two FFT windows instead of one (the "2–1" row in the left pane) yields to an accuracy difference of about 5%, i.e. it is better to use two windows instead of one. This difference is statistically significant since the error bars do not cross the dashed vertical line at zero. In contrast, there is no significant difference between using four versus two windows.

When interactions must be displayed, the plots become more crowded since there are generally a large number of possible pairwise comparisons between interacting factors. For clarity, such plots are sometimes omitted from the results presented herein, but we do take their results into account when performing, for example, model selection.

5.3. Statistical Graphics

We frequently use *box-whisker plots* to summarize empirical distributions. Our conventions for those plots follow the generally-employed ones (Tukey, 1977), and are illustrated in Fig. 6 (points considered as "outliers" are those whose distance to the "central box" is more than 1.5 times the height of the box).

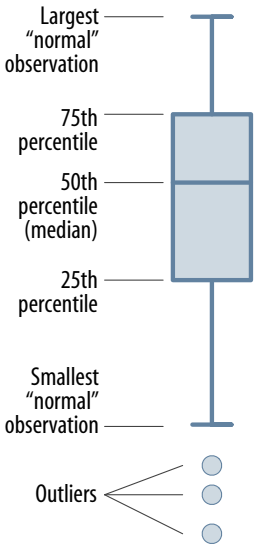

FIGURE 6. Illustration of a box plot as used in this paper.

6. EXPERIMENTS

In the context of the classification task we are interested in, it is sensible to assume a *uniform prior* on test classes: we want the classifier to be equally apt at recognizing all classes, regardless of imbalances that may arise due to limited training data. For this reason, all the results that we report in the following sections are *normalized accuracies*, which correct for the unequal class frequencies.

The normalization is carried out as follows. Let $N_{i,j}$ be the number of elements in row i and column j of the test confusion matrix (i.e. the number of examples of true class i that have been classified as class j by a model). We start by computing the class accuracies as the fraction of correctly classified examples of class i,

$$\widehat{\mathsf{Acc}}_i = \frac{N_{i,i}}{\sum_j N_{i,j}}.$$

We then average the individual class accuracies to obtain the normalized accuracy:

$$\widehat{\mathsf{Acc}} = \frac{1}{C}\sum_i \widehat{\mathsf{Acc}}_i, \tag{16}$$

where C is the total number of classes.

6.1. Results with REALISTIC-BY-DAY

6.1.1. *Support Vector Machines.* To give a flavor of the dramatic impact of preprocessing choice on performance, Fig. 7 illustrates the distribution of accuracies obtained by all experiments run with SVMs in the REALISTIC-BY-DAY setting. Each data point on this plot represents a test-accuracy result, and a kernel density estimator computed from these points is shown for the purpose of visualization. The figure separately examines two segmentations (`ApPeakNoTrunc` and `ARLTruncated`), several number of FFT windows and whether the logarithm of the FFT coefficients is taken.

We first note that the best performance is obtained with the `ApPeakNoTrunc` segmentation. However, performance is remarkably constant under the `ARLTruncated` segmentation, and exhibits a noticeable decrease only when taking a single FFT window. In contrast, the `ApPeakNoTrunc` segmentation interacts strongly with both the number of windows and taking the log: taking a longer portion of the signal can considerably improve performance, taking it from the 50's to the low 80's; likewise, for this segmentation, the additional normalization brought forth by the log is beneficial.

6.1.2. *Discriminative Restricted Boltzmann Machines.* To evaluate DRBMs in the REALISTIC-BY-DAY setting, we first analyze the influence of hyperparameters through the following ANOVA table. This table shows that all hyperparameters that we considered have a significant effect on performance, but also that interactions are important. Figure 8 summarizes the impact of main effects.

We separately analyzed interactions up to the third order between, respectively, the DRBM and preprocessing hyperparameters. Some of those interactions turn out to be significant, and for this reason, understanding regions of good performance in the space of hyperparameters is a bit involved. The DRBM generative learning weight has a

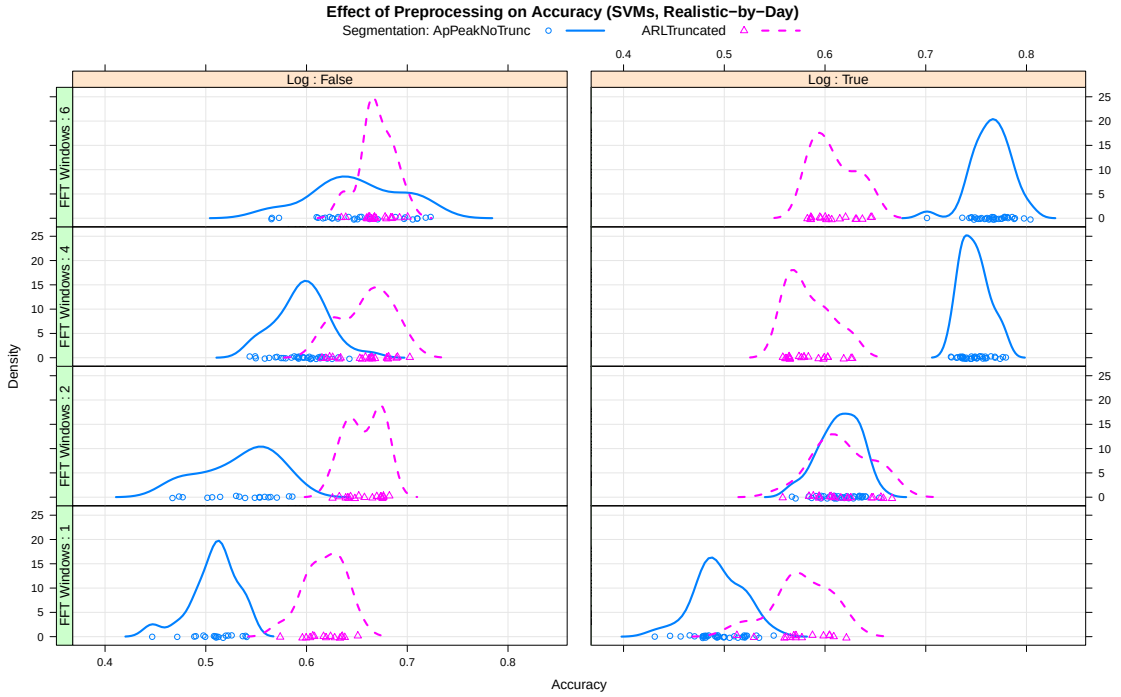

FIGURE 7. Effect of preprocessing on accuracy for SVMs, under the REALISTIC-BY-DAY setting. Accuracies appear on the x-axis, and a kernel density estimate is given in the y-axis; this representation illustrates the distributional effect on classification accuracy of an hyperparameter choice, *across all other hyperparameter choices* (which appear as individual points over which the distribution is plotted). The segmentation type (line styles), number of FFT windows (rows) and whether to take the logarithm of the FFT coefficients (columns) are independently varied. The semi-automatic segmentation `ApPeakNoTrunc` is clearly superior when considering the best of the other design choices (taking log-spectra with 6 FFT windows).

clear optimal value of 0.03 among the values tried. At this value, the effect of the number of hidden units (either 50 or 150) is not significant, nor is the learning rate (both 0.001 and 0.003 are equally good).[9] As to the preprocessing hyperparameters, the segmentation method `ApPeakNoTrunc` dominates the alternative `ARLTruncated`, and at this value, there is no significant difference between taking either 4 or 6 FFT windows (the two optimal values). Moreover, a window size of 256 is optimal at these settings.

```
                                  Df   Sum Sq  Mean Sq  F value    Pr(>F)
Weight of Generative Term          5   0.0403   0.0081   66.182 < 2.2e-16 ***
Discriminative Learning Rate       1   0.0011   0.0011    8.800 0.0037518 **
Number of Hidden Units             1   0.0015   0.0015   12.253 0.0006904 ***
Segmentation Type                  1   0.0685   0.0685  561.619 < 2.2e-16 ***
Number of FFT Windows              3   0.4132   0.1377 1129.877 < 2.2e-16 ***
FFT Window Duration                1   0.0296   0.0296  242.840 < 2.2e-16 ***
Wt Gen Term : Learn. Rate  Ix   5   0.0008   0.0002    1.348 0.2504596
Wt Gen Term : Nb Hidden    Ix   5   0.0014   0.0003    2.328 0.0479247 *
Learn. Rate : Nb Hidden    Ix   1   0.0012   0.0012    9.727 0.0023604 **
Segm Type : Nb FFT Windows Ix   3   0.3021   0.1007  826.043 < 2.2e-16 ***
Segm Type : FFT Window Dur Ix   1   0.0036   0.0036   29.803 3.377e-07 ***
Nb Windows: Window Dur Ix       1 3.33e-06 3.33e-06    0.027 0.8691031
Wt Gen Term : Learning Rate
            : Nb Hidden    Ix   5   0.0004   0.0001    0.602 0.6981798
Residuals                     102   0.0124   0.0001
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

[9] Interaction tables between these hyperparameters are rather large and are omitted for clarity.

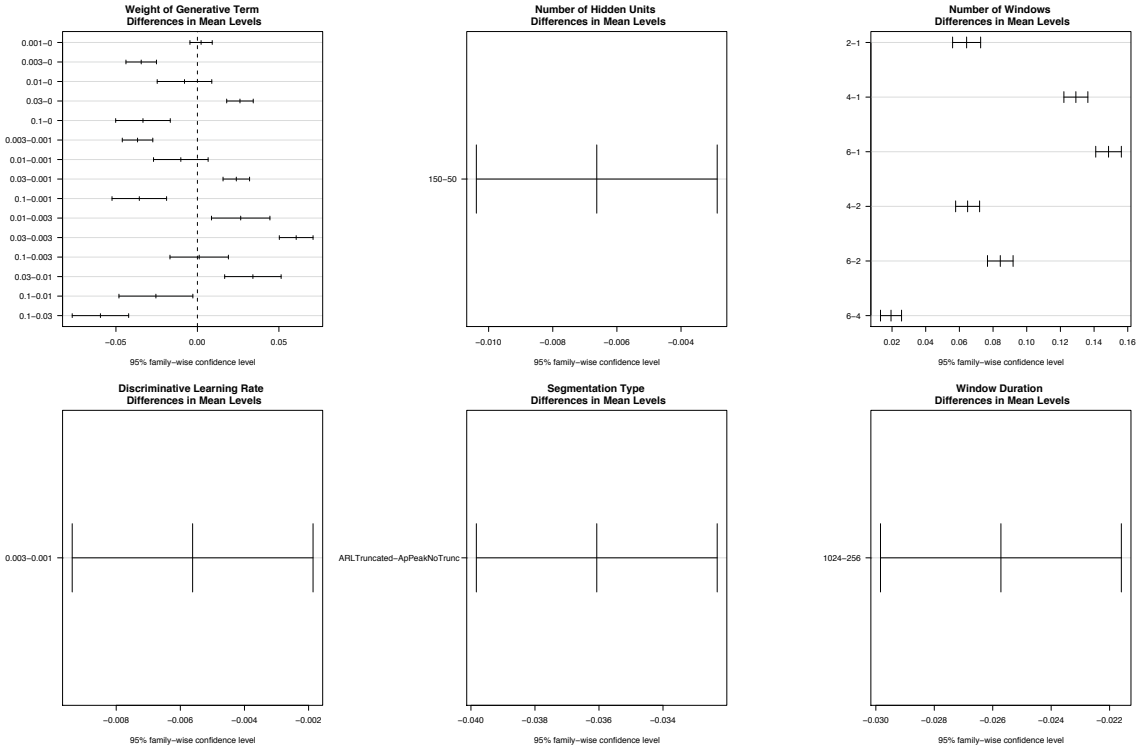

FIGURE 8. Impact of hyperparameters on classification accuracy for DRBMs, in the REALISTIC-BY-DAY scenario. Only main effects are included, but since several hyperparameter interactions are significant, direct conclusions about performance cannot be drawn solely from this plot; see the text for details.

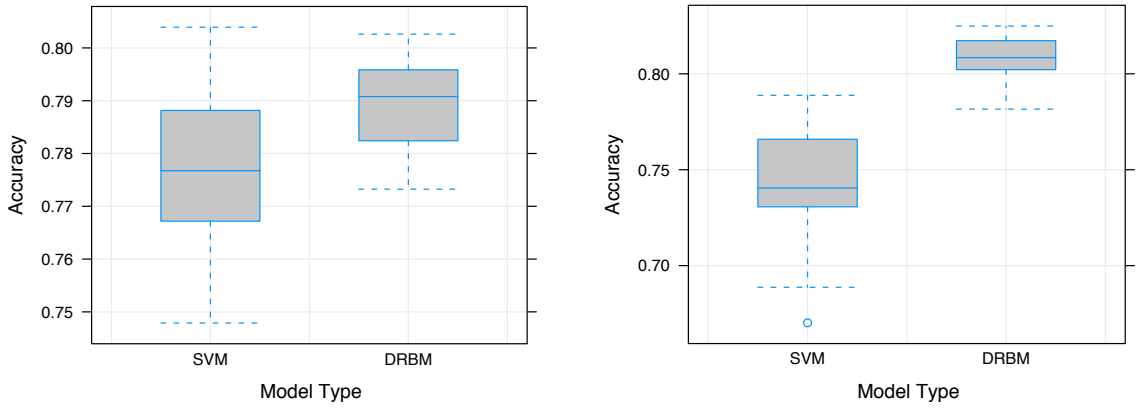

FIGURE 9. **Left:** Accuracies achieved by SVMs and DRBMs under the REALISTIC-BY-DAY setting. Statistically-indistinguishable hyperparameter values for each model type are included in this plot. In this realistic setting, DRBMs are superior to SVMs both in mean performance (statistically significant) and in terms of robustness (lower variance). **Right:** Accuracies achieved by SVMs and DRBMs under the REALISTIC-BY-RANGE setting.

6.1.3. *Comparison between SVM and DRBM on* REALISTIC-BY-DAY. Finally, we can restrict attention only to the respective "best" subsets of hyperparameters previously identified for SVMs and DRBMs (up to statistically identifiable differences through the ANOVAs, which were computed for the SVMs as well, even if we have not shown them here), and directly compare the two model types. Figure 9 (left) shows the distribution of test accuracies achieved by each model. Although the distributions overlap, the *mean accuracy* shows a slight advantage for DRBMs, at 78.2%, against 77.7% for SVMs. The difference is significant at the 90% level ($p = 0.08$).

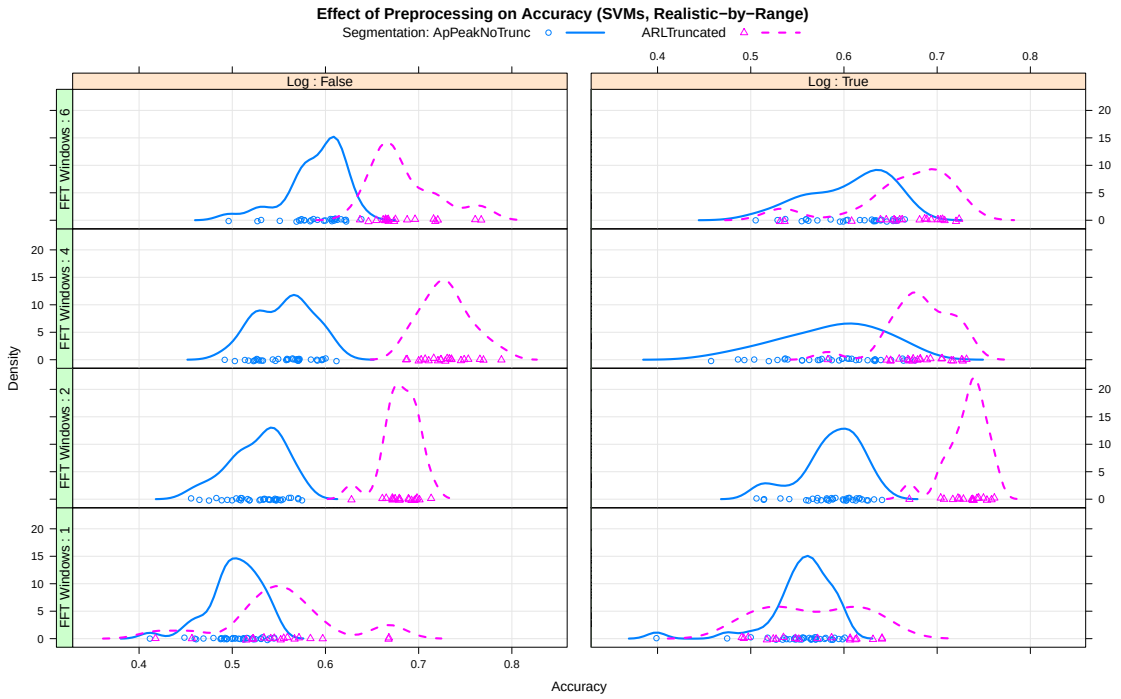

FIGURE 10. Effect of preprocessing on accuracy for SVMs, under the REALISTIC-BY-RANGE setting. The segmentation type (line styles), number of FFT windows (rows) and whether to take the logarithm of the FFT coefficients (columns) are independently varied.

6.2. Results with REALISTIC-BY-RANGE

6.2.1. *Support Vector Machines.* The impact of preprocessing choices on the accuracy of SVMs under the REALISTIC-BY-RANGE scenario is depicted in Fig. 10 (which should be compared to the same plot for the REALISTIC-BY-DAY setting in Fig. 7). We observe significant differences between the two settings. First, the best segmentation is now `ARLTruncated`. Moreover, taking the logarithm of the coefficients uniformly helps performance, and the number of FFT windows does not appear so important, as long as more than one window is taken.

6.2.2. *Discriminative Restricted Boltzmann Machines.* For DRBMs in the REALISTIC-BY-RANGE setting, the ANOVA table below shows that most hyperparameters are significant except the number of hidden units. The FFT window duration is also barely significant. A plot of the pairwise mean accuracy differences in the hyperparameter levels (main effects only) appears in Fig. 11.

Statistically significant interactions between hyperparameters makes choosing good-performing subsets slightly elaborate: the generative learning weight should be between 0.003 and 0.01, the learning rate between 0.001 and 0.01, with the `ARLTruncated` segmentation and 4 FFT windows.

```
                                Df  Sum Sq Mean Sq  F value    Pr(>F)
Weight of Generative Term        6 0.41470 0.06912  91.7878 < 2.2e-16 ***
Discriminative Learning Rate     3 0.11391 0.03797  50.4260 < 2.2e-16 ***
Number of Hidden Units           1 0.00003 0.00003   0.0367    0.8485
Segmentation Type                1 0.44070 0.44070 585.2551 < 2.2e-16 ***
Number of FFT Windows            3 0.56485 0.18828 250.0461 < 2.2e-16 ***
FFT Window Duration              1 0.00228 0.00228   3.0296    0.0849 .
Wt Gen Term : Learn. Rate  Ix    9 0.03714 0.00413   5.4806 4.378e-06 ***
Wt Gen Term : Nb Hidden    Ix    6 0.03068 0.00511   6.7913 4.830e-06 ***
Learn. Rate : Nb Hidden    Ix    3 0.00416 0.00139   1.8408    0.1448
Segm Type : Nb FFT Windows Ix    3 0.07140 0.02380  31.6080 2.230e-14 ***
Gt Gen Term : Learning Rate
            : Nb Hidden    Ix    9 0.00530 0.00059   0.7820    0.6333
Residuals                       98 0.07379 0.00075
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

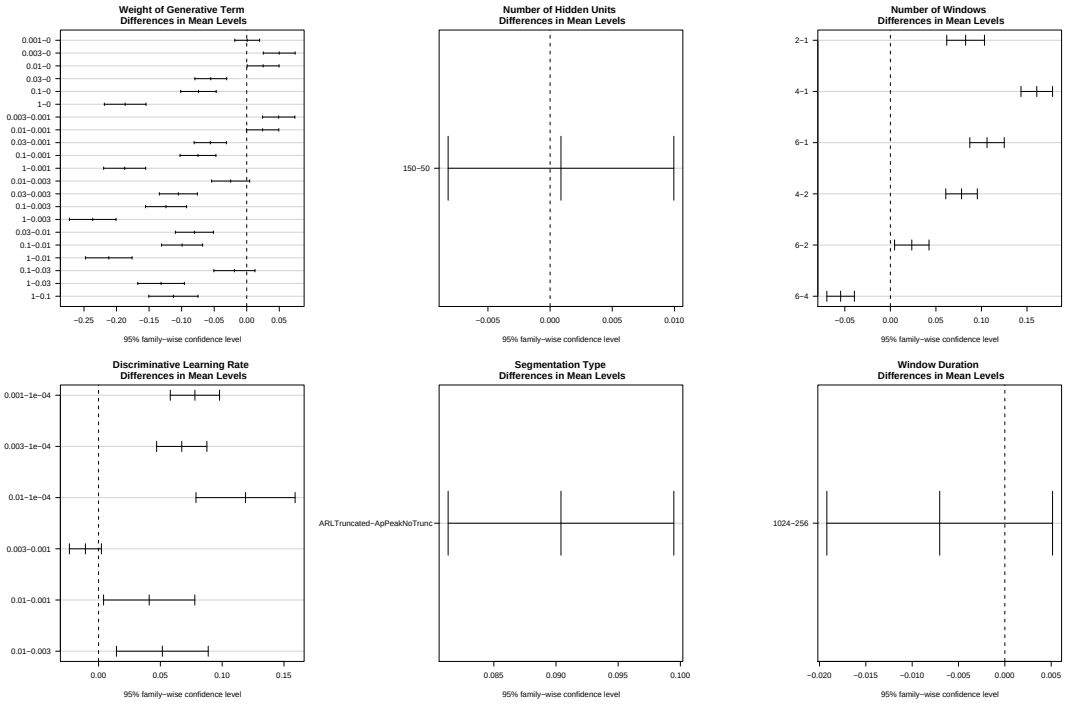

FIGURE 11. Impact of hyperparameters on classification accuracy for DRBMs, in the REALISTIC-BY-RANGE scenario. Only main effects are included, but since several hyperparameter interactions are significant, direct conclusions about performance cannot be drawn solely from this plot; see the text for details.

6.2.3. *Comparison between SVM and DRBM on Realistic-by-Range.* To conclude this investigation we restrict the hyperparameters of SVMs and DRBMs to their best-performing respective subsets (up to statistical comparability) to compare these two model types in the REALISTIC-BY-RANGE setting (again, we omit here the details of the statistical analysis for SMVs, in favor of the more visual density plot). Figure 9 (right) shows the distribution of test accuracies obtained by each model. In this setting, DRBMs appear to outperform SVMs by a substantial margin, the former reaching 80.8% mean accuracy and the latter 74.4%. This difference in mean accuracy is extremely statistically significant ($p < 10^{-6}$).

6.3. Are DRBMs Less Sensitive to Preprocessing Choice?

We have seen that the choice of preprocessing hyperparameters has a very significant impact on performance for SVMs in both the REALISTIC-BY-DAY and REALISTIC-BY-RANGE settings (Figures 7 and 10). Beyond raw accuracy results, this section tackles a different question and examines whether one model (among SVMs and DRBMs) exhibits more sensitivity to preprocessing choice than the other.

The analysis technique must be approached with care, since both methods involve very different model-specific hyperparameters. We are not quite interested in determining which is the more sensitive model at the best hyperparameter settings, but rather the expected sensitivity to preprocessing variations *for any fixed model-specific hyperparameter setting*.

In the spirit of ANOVA models, one relatively simple avenue is to attempt to explain the measured test accuracy for an experiment by a set of parameters that depend only on the combination of model-specific hyperparameters used for that experiment, and letting the remaining variations (due to preprocessor choice) be absorbed by the residuals. Let Acc_i be the test accuracy obtained by experiment i, and let $h(i)$ be an integer denoting the unique combination of hyperparameters used for this experiment. For instance, for an SVM, the possible hyperparameters are the choice of kernel (three possibilities), and the multiclass strategy (two possibilities). If we perform experiments with all six combinations, then $h(i)$ will be an integer between 1 and 6. Note that preprocessing hyperparameters are voluntarily excluded from these combinations because it is the preprocessing hyperparameters that are the subject of our inference procedure.

The resulting test accuracy for experiment i is represented as

$$\mathsf{Acc}_i = \beta_{h(i)} + \varepsilon_i, \tag{17}$$

where $\beta_{h(i)}$ represents the *mean accuracy* obtained by all experiments sharing hyperparameter combination $h(i)$. This representation is an instance of a so-called *random-effects model* in statistics (McCulloch *et al.*, 2008; Pinheiro and

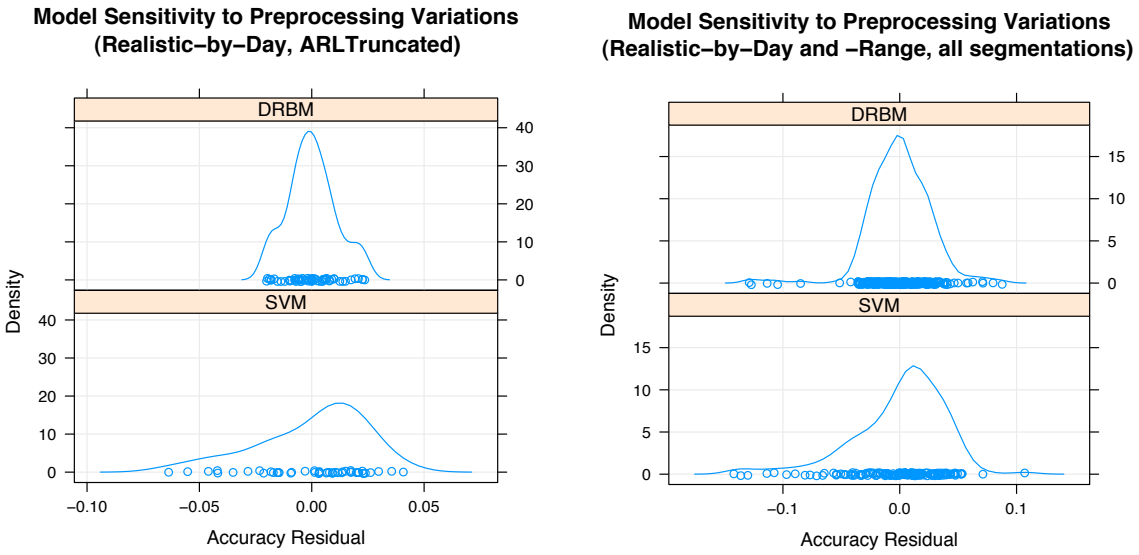

FIGURE 12. **Left:** Sensitivity of SVMs and DRBMs to the choice of preprocessing for the `ARLTruncated` segmentation, under the REALISTIC-BY-DAY setting. **Right:** Same results, across all segmentations and for both REALISTIC-BY-DAY and REALISTIC-BY-RANGE settings. DRBMs are seen to exhibit lower variance than SVMs, implying that they are less sensitive to the choice of preprocessing hyperparameters.

Bates, 2000), where parts of the model structure depends on characteristics of the test case (here the specific combination of hyperparameters pertaining to experiment i). These parameters are fit by simple linear regression.

Note that the regression residual ε_i absorbs all the variability in the accuracy Acc_i that is excluded from the main causes, specifically that due to the choice of preprocessor (since this is the only remaining source of variance, after we control for model-specific hyperparameters). Hence, and this is the core of the approach we follow here, we can test hypotheses about the distribution of those residuals, and if that of SVMs exhibits a greater variance than that of DRBMs, we can conclude the former are more affected by the choice of preprocessing hyperparameters.

Let $\boldsymbol{\varepsilon}_{SVM}$ and $\boldsymbol{\varepsilon}_{DRBM}$ respectively be the set of residuals belonging to experiments performed with SVMs and DRBMs. From introductory statistics, it is well known that under the null hypothesis of equality of variance and assuming that $\boldsymbol{\varepsilon}_{SVM}$ and $\boldsymbol{\varepsilon}_{DRBM}$ are both drawn from a normal distribution, the ratio

$$\frac{\mathrm{Var}[\boldsymbol{\varepsilon}_{SVM}]}{\mathrm{Var}[\boldsymbol{\varepsilon}_{DRBM}]}$$

is distributed according to Fisher's F distribution with P, Q Degrees Of Freedom (DOF), where P and Q are respectively the number of DOF in the numerator and denominator.[10] This forms the basis of the F-test used for comparing variances that is used here.

Restricting attention to the `ARLTruncated` segmentation under the REALISTIC-BY-DAY setting, we fit a model of the form (17) to experiment results, and plot the pattern of residuals in Fig. 12 (left). From inspection, the residuals of SVMs exhibit a much greater variance than those of DRBMs. This is confirmed formally by an F-test, which gives an extremely significant variance ratio of 5.19 (for SVMs in the numerator and DRBMs in the denominator), with 95% confidence intervals ranging from 2.93 to 9.47. Since the confidence interval does not include the point 1.0, we conclude that this ratio is significantly greater than one, implying that the performance of SVMs appears more affected by the choice of preprocessing in this context.

We would like to generalize this conclusion to other settings and segmentations, ideally through a joint test. A difficulty with a direct application of the model (17) is that individual settings and segmentations introduce a significant variance in and of themselves, and not controlling for those effects would result in a test losing all its statistical power. To work around this complication, we shall incorporate *fixed effects* in the previous random effects model, yielding a so-called *mixed-effects model*. In this context, the fixed effects are shared between SVMs and DRBMs and control for the following variables:

- Segmentation (either `ARLTruncated` or `ApPeakNoTrunc`),
- Evaluation setting (either REALISTIC-BY-DAY or REALISTIC-BY-RANGE),
- Number of FFT windows.

[10] Roughly speaking, the number of DOF in the regression residuals is computed as the number of observations in the training set minus the number of parameters that are part of the regression model.

TABLE 1. Summary of the best accuracy results obtained for the detonation main-type classification problem, for each evaluation setting and model type. The $\pm$ denote 95% confidence intervals.

		REALISTIC-BY-DAY	REALISTIC-BY-RANGE
Best	SVMs	$80.4\% \pm 1.4\%$	$84.2\% \pm 1.3\%$
Results	DRBMs	$82.7\% \pm 1.3\%$	$83.0\% \pm 1.3\%$
Median	SVMs	$77.7\% \pm 1.4\%$	$74.0\% \pm 1.5\%$
Results	DRBMs	$79.1\% \pm 1.4\%$	$80.8\% \pm 1.4\%$

Although the latter factor is technically part of preprocessing, we have seen (e.g. Fig. 7) that in some circumstances its choice is so important as to dwarf the impact of all other hyperparameters. As we shall see, our conclusions are no less diminished by inclusion of this factor.

Let $p(i)$ be an integer denoting the specific combination of the three above variables used for experiment i, and $h(i)$ denoting the specific combination of model hyperparameters, as previously. The mixed-effects model that we consider is specified as

$$\mathsf{Acc}_i = \alpha_{p(i)} + \beta_{h(i)} + \varepsilon_i,$$

where $\alpha_{p(i)}$ is the fixed effects part (with parameters shared between SVMs and DRBMs) controlling for the previous two "setting" variables, $\beta_{h(i)}$ is the random effects part controlling for model hyperparameters, and ε_i is a residual absorbing unmodeled causes (i.e. the remaining preprocessing hyperparameters). Fitting models of this type is, in general, more involved than simple linear regression and is usually performed by *restricted maximum likelihood* (REML) estimation (McCulloch *et al.*, 2008).

However, analysis of the residuals can proceed as previously. Figure 12 (right) displays the obtained residuals under the mixed-effects model for all experiments, for both SVMs and DRBMs. Although the difference is less striking than before, the observed variance ratio is nevertheless of 2.16, with a 95% confidence interval between 1.65 and 2.86 (under the hypotheses of the F-test). Again, since the interval is beyond the value 1.0, we reject the null of equality of variance. We conclude that there is significant evidence in favor of the proposition that DRBMs exhibit less variability to the choice of preprocessing than SVMs across segmentations and evaluation settings.

6.4. Summary of Results

This section presented many experimental results on the detonation main-type classification problem. A summary of our best models (for both SVMs and DRBMs) for each of the two evaluation settings appears in Table 1. This table also contains the median performance obtained after selecting the statistically-comparable "best" subset of hyperparameters (following the methodology outlined in 3.2).

7. CONCLUSION AND FUTURE WORK

In this paper we highlighted the challenges of a detonation type classification task where one must differentiate betweeen launches of MORTARs, ROCKETs and RPGs. We described how to properly train and evaluate classifiers so as to be able to perform model selection and estimate their generalization ability.

We applied our methodology to compare the recently proposed Discriminative Restricted Boltzmann Machine (DRBM) to the Support Vector Machine (SVM). Although both SVMs and DRBMs exhibit comparable final performance when selecting the best models, we observe that DRBMs are slightly superior overall, and are less sensitive to the choice of preprocessing hyperparameters than SVMs. This makes them particularly appealing classification tools for such a task where model selection is difficult.

Until now we have been relying on manual segmentation of audio signals. This is impractical for a realistic system deployed on the field. We are currently experimenting with automatic segmentation techniques based on the edge detector algorithm that we used in this study to assist manual segmentation (see Section 2.2).

8. ACKNOWLEDGEMENTS

This work was supported by the US Army Research Laboratory, Acoustic & EM Sensing Branch, contract no. W911NF-07-D-0001.

REFERENCES

Bedard, J. and Pare, S. (2003). Ferret, a small arms' fire detection system: Localization concepts. In *SPIE Proceedings of Sensors, and Command, Control, Communications and Intelligence (C3I) Technologies for Homeland Defense and Law Enforcement*, volume 5071, pages 497–509.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, **2**(1), 1–127.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy layer-wise training of deep networks. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, Cambridge, MA.

Bogert, B. P., Healy, M. J. R., and Tukey, J. W. (1963). The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking. In M. Rosenblatt, editor, *Proceedings of the Symposium on Time Series Analysis*, pages 209–243, New York, NY. John Wiley & Sons.

Bouchard, G. and Triggs, B. (2004). The tradeoff between generative and discriminative classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, pages 721–728, Prague.

Box, G. E., Hunter, W. G., and Hunter, J. S. (2005). *Statistics for Experimenters: Design, Innovation, and Discovery*. John Wiley & Sons, second edition.

Carreira-Perpiñán, M. A. and Hinton, G. (2005). On contrastive divergence learning. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, Jan 6-8, 2005, Savannah Hotel, Barbados*, pages 33–40. Society for Artificial Intelligence and Statistics. (Available electronically at http://www.gatsby.ucl.ac.uk/aistats/).

Chang, C.-C. and Lin, C.-J. (2001). *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.

Childers, D. G., Skinner, D. P., and Kemerait, R. C. (1977). The cepstrum: A guide to processing. *Proceedings of the IEEE*, **65**(10), 1428–1443.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, **20**(3), 273–297.

Croarkin, C. and Tobias, P., editors (2006). *NIST/Sematech e-Handbook of Statistical Methods*. U.S. Commerce Department. Available at `http://www.itl.nist.gov/div898/handbook/index.htm`.

Deller, J. R., Hansen, J. H., and Proakis, J. G. (1999). *Discrete-Time Processing of Speech Signals*. John Wiley & Sons / IEEE Press.

Desai, S., Hohil, M., and Morcos, A. (2006). Classifying launch/impact events of mortar and artillery rounds utilizing DWT-derived features and feedforward neural networks. *Independent Component Analyses, Wavelets, Unsupervised Smart Sensors, and Neural Networks IV*, **6247**(1).

Desai, S., Hohil, M., and Morcos, A. (2007). Utilizing distinguishable acoustic features for variant discrimination. In *19th International Congress on Acoustics*.

Engelberg, S. (2008). Edge detection using Fourier coefficients. *American Mathematical Monthly*, **115**(6), 499–513.

Fan, R., Chen, P., and Lin, C.-J. (2005). Working set selection using second order information for training SVM. *Journal of Machine Learning Research*, **6**, 1889–1918.

Graps, A. (1995). An introduction to wavelets. *Computing in Science and Engineering*, **2**, 50–61.

Grosse, R., Raina, R., Kwong, H., and Ng, A. Y. (2007). Shift-invariant sparse coding for audio classification. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, **14**, 2002.

Hinton, G. E. (2007). To recognize shapes, first learn to generate images. In P. Cisek, T. Drew, and J. Kalaska, editors, *Computational Neuroscience: Theoretical Insights into Brain Function*. Elsevier.

Hinton, G. E., Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief networks. *Neural Computation*, **18**(7), 1527–1554.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Trans. Neural Networks*, **13**(5), 415–425.

Hsu, J. (1996). *Multiple Comparisons: Theory and Methods*. Chapman & Hall/CRC, Boca Raton, FL.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, **33**, 82–95.

Larochelle, H. and Bengio, Y. (2008). Classification using discriminative restricted Boltzmann machines. In A. McCallum and S. Roweis, editors, *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pages 536–543. Omnipress.

Lasserre, J. A., Bishop, C. M., and Minka, T. P. (2006). Principled hybrids of generative and discriminative models. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 87–94, Washington, DC, USA. IEEE Computer Society.

Lee, H., Pham, P., Largman, Y., and Ng, A. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1096–1104.

Marin-Jimenez, M., de la Blanca, N. P., Mendoza, M., Lucena, M., and Fuertes, J. (2009). Learning action descriptors

for recognition. *International Workshop on Image Analysis for Multimedia Interactive Services*, pages 5–8.

McCulloch, C. E., Searle, S. R., and Neuhaus, J. M. (2008). *Generalized, Linear, and Mixed Models*. Wiley Series in Probability and Statistics. John Wiley & Sons, Hoboken, NJ, second edition.

Morcos, A., Grasing, D., and Desai, S. (2008). Artillery/mortar type classification based on detected acoustic transients. *Unattended Ground, Sea, and Air Sensor Technologies and Applications X*, **6963**(1).

Naz, P. and Marty, C. (2006). Sound detection and localization of small arms, mortars, and artillery guns. *Unattended Ground, Sea, and Air Sensor Technologies and Applications VIII*, **6231**(1).

Ng, A. Y. and Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 841–848, Cambridge, MA. MIT Press.

Pinheiro, J. C. and Bates, D. M. (2000). *Mixed Effects Models in S and S-Plus*. Springer-Verlag, New York, NY.

Quatieri, T. F. (2001). *Distrete-Time Speech Signal Processing: Principles and Practice*. Prentice Hall.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press.

Salakhutdinov, R., Minh, A., and Hinton, G. (2007). Restricted Boltzmann machines for collaborative filtering. In Z. Ghahramani, editor, *Proceedings of the 24th Annual International Conference on Machine Learning (ICML 2007)*, pages 791–798. Omnipress.

Schölkopf, B., Mika, S., J.C.Burges, C., Knirsch, P., Mller, K.-R., Rtsch, G., and J. Smola, A. (1999). Input space versus feature space. *IEEE Trans. Neural Networks*, **100**(5), 1000–1017.

Smith, B. (2006). Feature extraction for transient acoustic event classification. University of Washington.

Tieleman, T. (2008). Training restricted Boltzmann machines using approximations to the likelihood gradient. In A. McCallum and S. Roweis, editors, *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, pages 1064–1071. Omnipress.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley, Reading, MA.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley & Sons.

Welling, M., Rosen-Zvi, M., and Hinton, G. (2005). Exponential family harmoniums with an application to information retrieval. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1481–1488. MIT Press, Cambridge, MA.