# Statistical Machine Learning Algorithms for Target Classification from Acoustic Signature

July 2009

Vincent Mirelli
U.S. Army Research Laboratory
2800 Powder Mill Road
Adelphi, MD, 20873-1197

Stephen Tenney
US Army Reseach Laboratory
2800 Powder Mill Road
Adelphi, MD, 20873-1197

Yoshua Bengio
University of Montreal
P.O. Box 6128, succ. Centre-Ville
Montreal, QC, H3C 3J7, Canada

Nicolas Chapados, Olivier Delalleau
ApSTAT Technologies
4200 Boul. St-Laurent, suite 408
Montreal, QC, H2W 2R2, Canada

**Abstract**

Machine learning classification algorithms are relevant to a large number of Army classification problems, including the determination of a weapon class from a detonation acoustic signature. However, much such work has been focused on classification of events from small weapons used for asymmetric warfare, which have been of importance in recent years. In this work we consider classification of very different weapon classes, such as mortar, rockets and RPGs, which are difficult to reliably classify with standard techniques since they tend to have similar acoustic signatures. To address this problem, we compare two recently-introduced state-of-the-art machine learning algorithms, Support Vector Machines and Discriminative Restricted Boltzmann Machines, and develop how to use them to solve this difficult acoustic classification task. We obtain classification accuracy results that could make these techniques suitable for fielding on autonomous devices. Discriminative Restricted Boltzmann Machines appear to yield slightly better accuracy than Support Vector Machines, and are less sensitive to the choice of signal preprocessing and model hyperparameters. Importantly, we also address methodological issues that one faces in order to rigorously compare several classifiers on limited data collected from field trials; these questions are of significance to any application of machine learning methods to Army problems.

## 1 Introduction

The objective of this work is to develop advanced signal processing and machine learning algorithms for performing automatic target classification of impulsive sources (such as detonations) that perform well when deployed in the field. In particular, we consider the task of discriminating launch signals generated by three weapon classes: `MORTAR`, `ROCKET`, and rocket-propelled grenades (`RPG`s).

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 proving grounds (APG, Dahlgreen, Yuma), which makes the evaluation of classifier reliability considerably more difficult. This data is described in more details in Section 2.1.

One of our contributions to the research on detonation classifiers regards statistical and methodological issues related to the problem of data scarcity: How do we fairly evaluate performance? How do we compare different classifiers in statistically trustworthy ways? In this paper 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.

We compare two statistical machine learning classifiers applied to our classification task: Support Vector Machines (SVM: a classical non-parametric discriminant classifier), and Discriminative Restricted Boltzmann Machines (DRBM: a recently proposed hybrid that combines a discriminant criterion and a generative criterion). *Discriminant classifiers*, such as SVMs, focus on learning the classification decision. In contrast, *generative classifiers* try to capture the actual distribution of signals. Potential advantages of generative classifiers include the ability to take advantage of unlabeled data, e.g., 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, Osindero, and Teh 2006; Bengio, Lamblin, Popovici, and Larochelle 2007; Ranzato, Poultney, Chopra, and LeCun 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.

Our experiments are performed with these classifiers on the classification task of discriminating between three main types of weapons studied here: `ROCKET`, `RPG` and `MORTAR` explosions. The experiments explore interactions of classifier choice with various segmentation and preprocessing choices. We find that although both SVMs and DRBMs can achieve comparable *best* performance, DRBMs perform slightly better, and have the significant advantage of being less sensitive to the choice of preprocessing and algorithm hyperparameters. Since model evaluation is difficult for this task due to the specificities of the data (as detailed in Section 2.1), such a conclusion indicates that DRBMs are particularly interesting in such a setting.

**Organization of this Paper** This paper is organized as follows: in Section 2 we provide an overview of the data and signal processing techniques employed; we follow in Section 3 by an overview of the methodological issues for a fair comparison between classification algorithms; in Section 4 we review the formulation of the SVM and DRBM algorithms employed, followed by a presentation of experimental results in Section 5. Finally, Section 6 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.

From the various data sources, we have built a dataset containing the acoustic signatures of all unique signatures of launches for weapons of types `MORTAR`, `ROCKET` and `RPG`. After a manual screening of the data to remove low-quality samples, this dataset contains a total of 636 different launching signatures, each having four individual signals (from the four microphones of a tetrahedral array). 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 other 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.

This 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 Dahlgren, 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, APG, 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 needs to take into account the fact that some of the launch events are represented by more than one signature. In most cases, this is because there is more than one tetrahedral array recording all of the events from an exercise, and therefore we get up to one signature per array per launch event. This can be important when designing an experimental setup meant to evaluate the performance of a model on new explosions, a topic that is discussed in Section 3.1.

### 2.2 Signal Segmentation

As the experiments presented in Section 5 will illustrate, the segmentation can have a significant impact on classifier performance. Two manual segmentations are available for the signals we consider: one performed by the ARL staff, and one performed by the ApSTAT staff. They are called respectively `ARLTruncated` and `ApPeakTruncated` in this paper. When we do not use the end of the segmentation (i.e. the signal is not right-truncated), they lead to the `ARLNoTrunc` and `ApPeakNoTrunc` segmentations.

### 2.3 Preprocessing by Spectral Features

Our preprocessing consists of spectral features obtained from the signal Fast Fourier Transform (FFT). We extract 35 spectral coefficients on frames starting according to a segmentation policy (described below). 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 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 (Deller, Hansen, and Proakis 1999; Quatieri 2001), since our 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 usually kept constant at 128. When 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). Additional results with this variant are reported in Section 5.

## 3 Methodological Issues

### 3.1 Correctness Issues and Data Splits

A classifier is meant to be used in the field, on new data which at the time of developing the classifier are not available, 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. 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, that 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 were really varying for each new day of recording explosions. 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.

### 3.2 Hyperparameter Selection

A rigorous procedure for hyperparameter selection would rely on *two-level cross-validation*. Unfortunately, given the very limited amounts of data in some of the classes that we must classify, we had to rely on a different procedure since two-level cross-validation may result in severe instability: due to the splitting constraints, instances from the minority classes can be nearly (or completely) absent from some 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 purposes of model comparison:

- On test results arising from a one-level cross-validation (performed in one of the two settings described in 3.1), we carry out an ANalysis Of VAriance (ANOVA) to determine statistically-significant main effects and interactions (Box, Hunter, and Hunter 2005; Croarkin and Tobias 2006).
- 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.
- Comparisons between models are performed by contrasting aspects of their respective accuracy distribution, for instance its mean, median or variance.

## 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 thus we only provide here the details specific to our experiment setting.

In particular, 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.[1] We considered in our experiments three widely used families of vector kernels: linear, polynomial and Radial Basis Function (RBF). The hyperparameters specific to each kernel are automatically chosen based on an "internal" three-fold cross-validation on the training set.

In their basic form, SVMs are fundamentally binary classifiers, so several methods have been proposed to generalize them to more than two classes. The two main options are one-against-all and one-against-one (Hsu and Lin 2002). In our experiments, both strategies performed almost the same.

### 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. Though such models are most often trained to only model the inputs of a classification task, they can also model the joint distribution of the inputs and associated target classes (e.g. as in Larochelle and Bengio (2008) and Tieleman (2008), or in the last layer of a deep neural network as in Hinton, Osindero, and Teh (2006)). In this paper, we present experiments with such a joint model, which is depicted in Figure 1.

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

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

[1] 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.

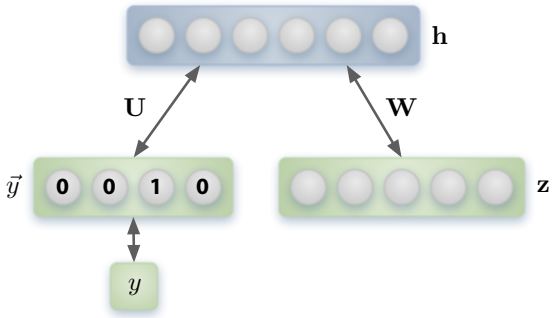


**Figure 1**: Restricted Boltzmann Machine modeling the joint distribution of inputs $\mathbf{z}$ and target class $y$ (represented as a one-hot vector by $\vec{y}$). The current state of the hidden units is labeled by $\mathbf{h}$.

where

$$E(y,\mathbf{z},\mathbf{h}) \quad = \quad -\mathbf{h}'\mathbf{W}\mathbf{z} - \mathbf{b}'\mathbf{z} - \mathbf{c}'\mathbf{h} - \mathbf{d}'\vec{y} - \mathbf{h}'\mathbf{U}\vec{y} \tag{1}$$

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, Rosen-Zvi, and Hinton 2005). We discuss later in this section how such a modification applies to our experiments.

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:

$$p(\mathbf{z}|\mathbf{h}) \quad = \quad \prod_i p(\mathbf{z}_i|\mathbf{h})$$

$$p(\mathbf{z}_i = 1|\mathbf{h}) \quad = \quad \text{sigm}\left(\mathbf{b}_i + \sum_j \mathbf{W}_{ji}\mathbf{h}_j\right) \tag{2}$$

$$p(y|\mathbf{h}) \quad = \quad \frac{e^{\mathbf{d}_y + \sum_j \mathbf{U}_{jy}\mathbf{h}_j}}{\sum_{y^*} e^{\mathbf{d}_{y^*} + \sum_j \mathbf{U}_{jy^*}\mathbf{h}_j}} \tag{3}$$

where $\text{sigm}(\cdot)$ is the logistic sigmoid. Equations 2 and 3 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:

$$p(\mathbf{h}|y,\mathbf{z}) \quad = \quad \prod_j p(\mathbf{h}_j|y,\mathbf{z})$$

$$p(\mathbf{h}_j = 1|y,\mathbf{z}) \quad = \quad \text{sigm}\left(\mathbf{c}_j + \mathbf{U}_{jy} + \sum_i \mathbf{W}_{ji}\mathbf{z}_i\right). \tag{4}$$

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:

$$\mathcal{L}_{gen} = -\sum_{i=1}^{N_{\mathcal{T}}} \log p(y^{(i)}, \mathbf{z}^{(i)}). \tag{5}$$

**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 $\lambda$
{ Notation: $a \leftarrow b$ means $a$ is set to value $b$
$a \sim p$ means $a$ is sampled from $p$}

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

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

{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**

In order to minimize the negative log-likelihood (eq. 5), 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 gradient (Hinton 2002). This approximation replaces the 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 $\lambda$.

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, Minh, and Hinton (2007), for reasonable numbers of classes $C$ (over which we must sum), this conditional distribution can be computed exactly and efficiently, by writing it as follows:

$$p(y|\mathbf{z}) = \frac{e^{\mathbf{d}_y}\prod_{j=1}^{H}\left(1 + e^{\mathbf{c}_j + \mathbf{U}_{j,y} + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i}\right)}{\sum_{y^*} e^{\mathbf{d}_{y^*}}\prod_{j=1}^{H}\left(1 + e^{\mathbf{c}_j + \mathbf{U}_{j,y^*} + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i}\right)}. \quad (6)$$

Precomputing the terms $\mathbf{c}_j + \sum_i \mathbf{W}_{j,i}\mathbf{z}_i$ neurons 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:

$$\mathcal{L}_{disc}(\mathcal{T}) = -\sum_{i=1}^{N_{\mathcal{T}}} \log p(y^{(i)}|\mathbf{z}^{(i)}). \tag{7}$$

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:

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

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 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, Bishop, and Minka (2006), it proved useful here and elsewhere (Bouchard and Triggs 2004). In this work, we used the following criterion:

$$\mathcal{L}_{hybrid}(\mathcal{T}) = \mathcal{L}_{disc}(\mathcal{T}) + \alpha \mathcal{L}_{gen}(\mathcal{T}) \tag{8}$$

where the weight of the generative criterion is controlled by $\alpha$. 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 $\alpha$ 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,[1] we used **Gaussian visible units** (Welling, Rosen-Zvi, and Hinton 2005; Bengio, Lamblin, Popovici, and Larochelle 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 1. From this new energy function, we can show that each conditional distribution $p(\mathbf{z}_i|\mathbf{h})$ is now a Gaussian distribution of mean $\mu_i$ and variance parameter $\sigma_i^2$:

$$\mu_i = \frac{\mathbf{b}_i + \mathbf{W}'_{\cdot,i}\mathbf{h}}{2\mathbf{a}_i^2}, \qquad \sigma_i^2 = \frac{1}{2\mathbf{a}_i^2}, \quad \forall i \in \{1,\ldots,d\}, \tag{9}$$

while the other conditional distributions of Equations (3), (4) and (6) 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 $\mathcal{T}^{\frac{4}{5}}$.

[1] We usually normalize the inputs in the $[-1,1]$ range.

## 5 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 data. For this reason, all the results that we report in the following sections are *normalized accuracies*, which correct for the unequal class frequencies.

### 5.1 Results with Realistic-by-Day

#### 5.1.1 Support Vector Machines

To give a flavor of the dramatic impact of preprocessing choice on performance, Fig. 2 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. 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 automatic segmentation of `ApPeakNoTrunc`. 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.

#### 5.1.2 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 3 summarizes the impact of main effects.

We separately analysed 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 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).[1] 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.

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

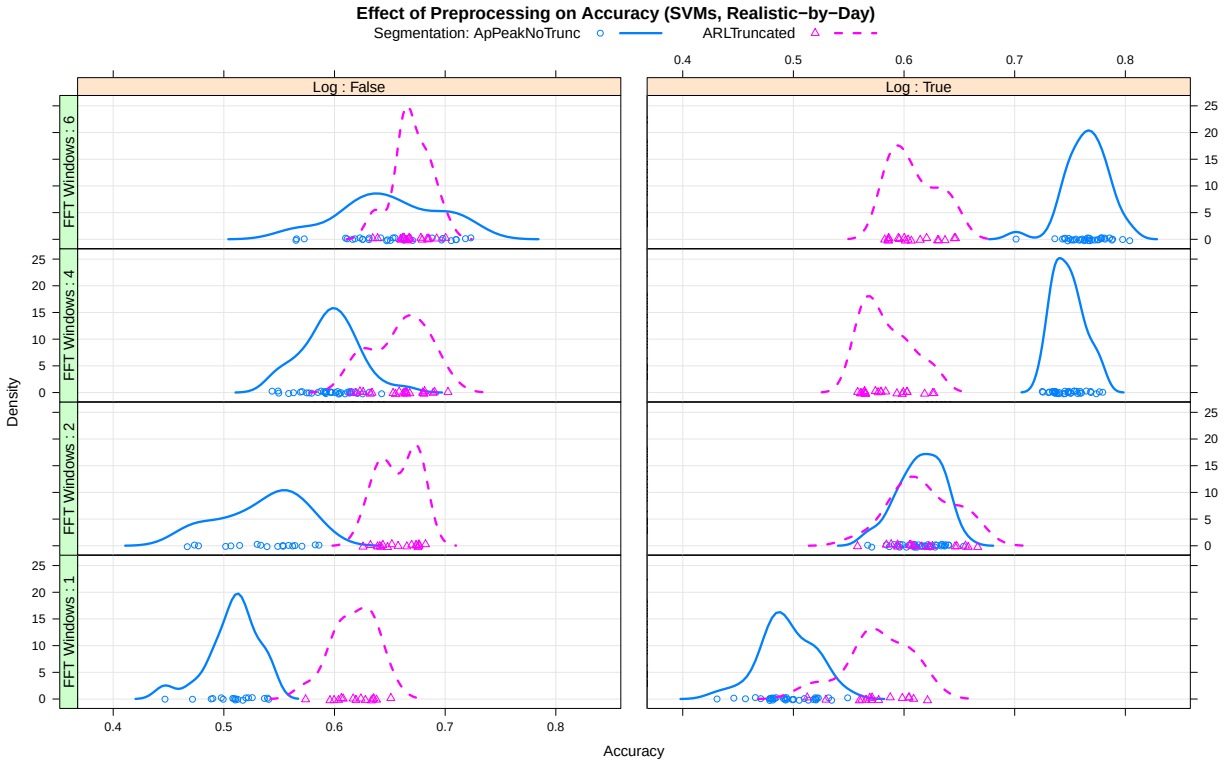


**Figure 2**: Effect of preprocessing on accuracy for SVMs, under the REALISTIC-BY-DAY setting. Accuracies appear on the $x$-axis, and a kernel density estimate is given in the $y$-axis; this representation illustrates the distributional effect on classification accuracy of an hyperparameter choice, *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 and whether to take the logarithm of the FFT coefficients are independently varied. The automatic segmentation (`ApPeakNoTrunc`) is clearly superior when considering the best of the other design choices (taking log-spectra with 6 FFT windows).

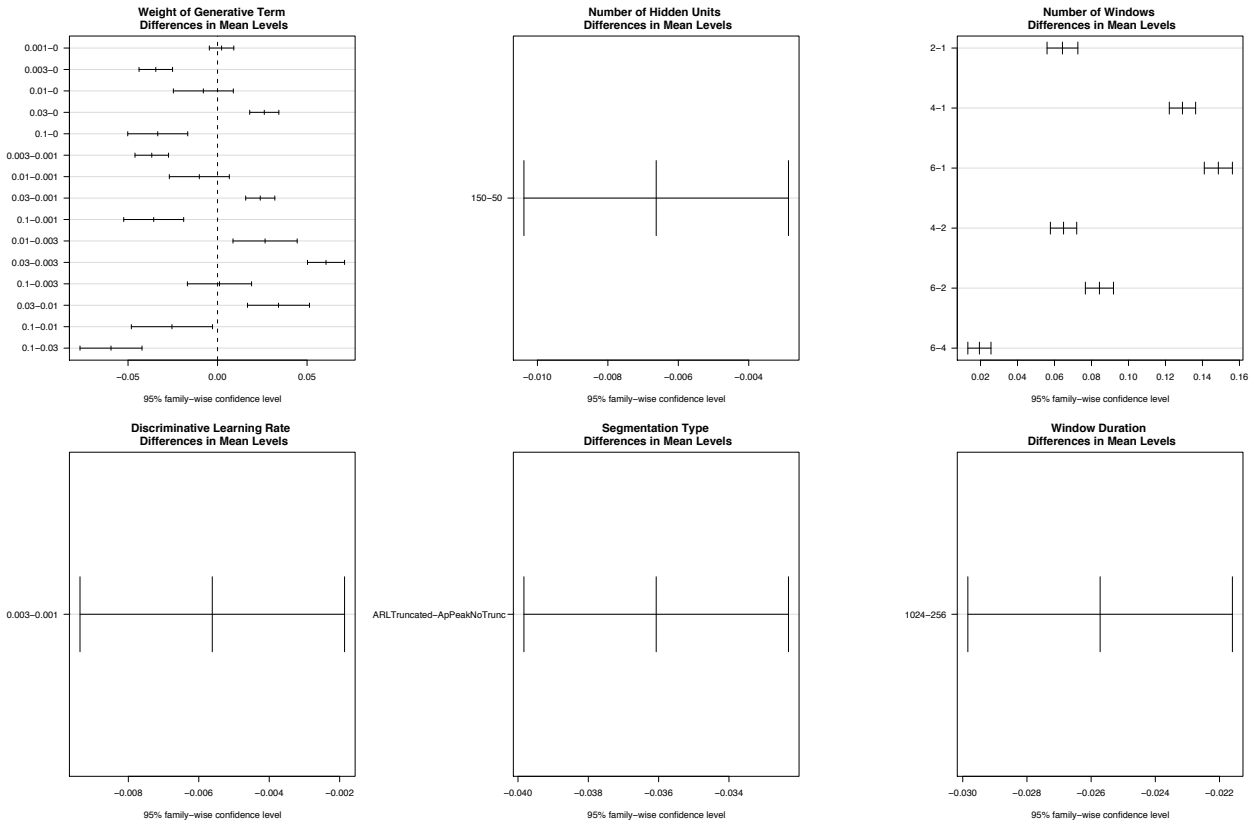


**Figure 3**: Impact of hyperparameters on classification accuracy for DRBMs, in the REALISTIC-BY-DAY scenario. Only main effects are included, but since several hyperparameter interactions are significant, direct conclusions about performance cannot be drawn solely from this plot; see the text for details.

| | 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

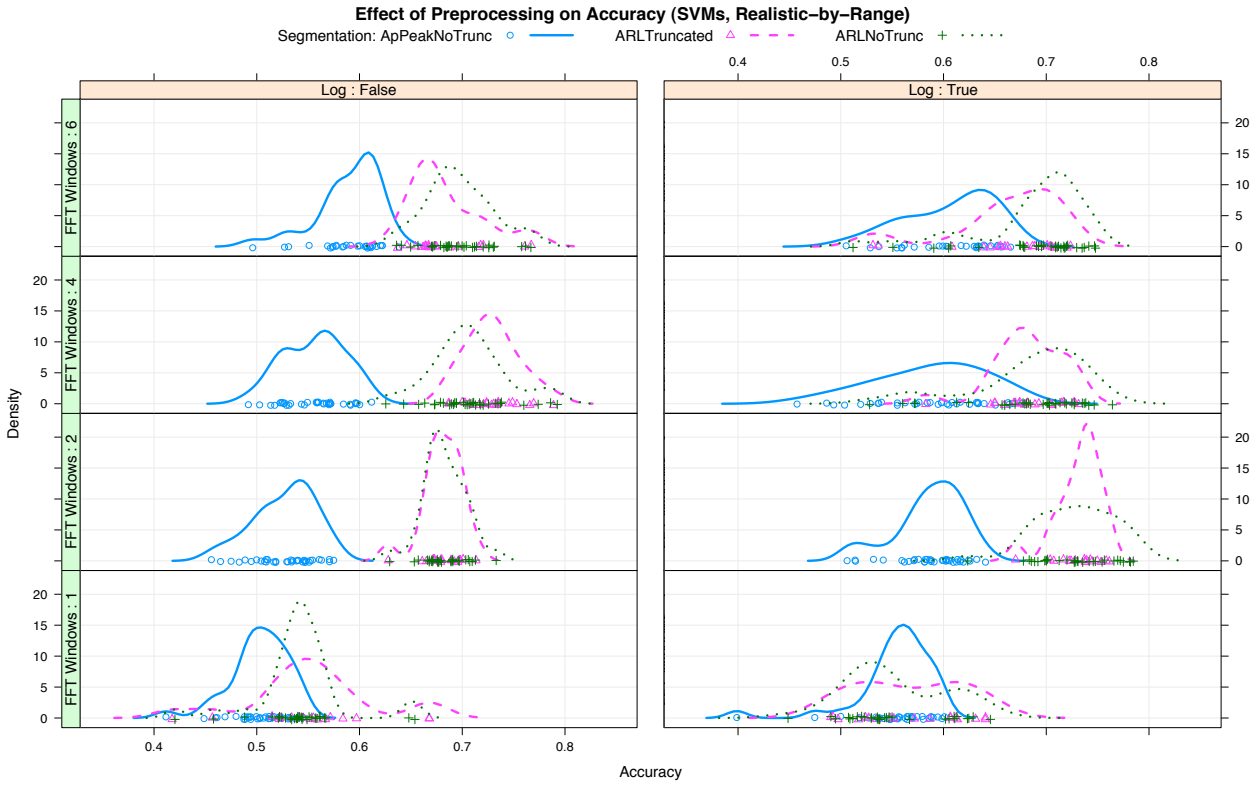


**Figure 4**: Effect of preprocessing on accuracy for SVMs, under the REALISTIC-BY-RANGE setting. The segmentation type (line styles), number of FFT windows and whether to take the logarithm of the FFT coefficients are independently varied.

### 5.1.3 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), and directly compare the two model types. Figure 6 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 ($p = 0.08$).

## 5.2 Results with Realistic-by-Range

### 5.2.1 Support Vector Machines

The impact of preprocessing choices on the accuracy of SVMs under the REALISTIC-BY-RANGE scenario is depicted in Fig. 4. (This should be compared to the same plot for the REALISTIC-BY-DAY setting in Fig. 2.) We observe significant differences between the two settings. First, different segmentations are now preferable (either `ARLTruncated` or `ARLNoTrunc` rather than `ApPeakNoTrunc`), and since the first two differ only by their endpoints but exhibit comparable accuracy, we conclude that the start of the segmentation is what matters most in this setting. Moreover, taking the logarithm of the coefficients uniformly helps performance, and the number of FFT windows does not appear so important, so long as more than one window is taken.

### 5.2.2 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. 5.

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
```

### 5.2.3 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. Figure 6 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 ($p < 10^{-6}$).

## 5.3 Are DRBMs Less Sensitive to Preprocessing Choice?

We have seen that the choice of preprocessing hyperparameters has a very significant impact on performance in both the REALISTIC-BY-DAY and REALISTIC-BY-RANGE settings (e.g. Figure 2 and 4). 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)

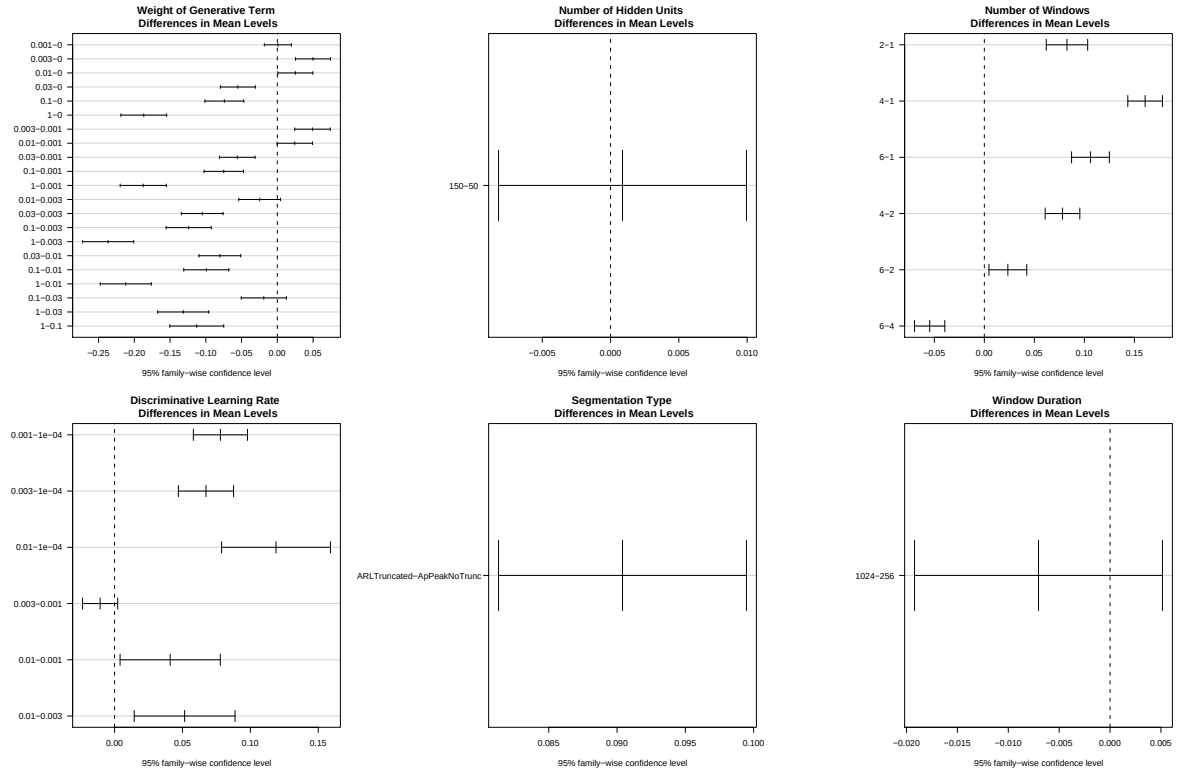


**Figure 5**: Impact of hyperparameters on classification accuracy for DRBMs, in the REALISTIC-BY-RANGE scenario. Only main effects are included, but since several hyperparameter interactions are significant, direct conclusions about performance cannot be drawn solely from this plot; see the text for details.

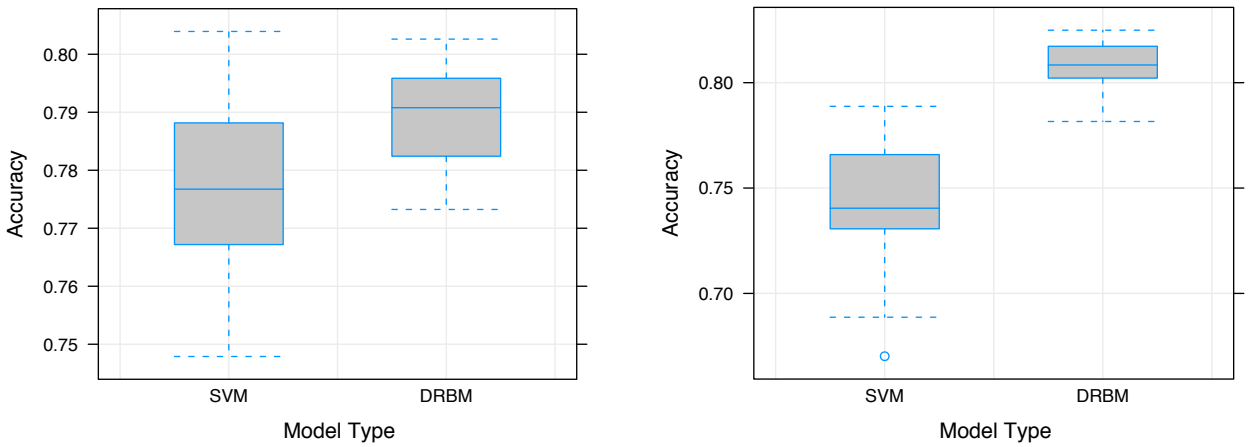


**Figure 6**: **Left:** Accuracies achieved by SVMs and DRBMs under the REALISTIC-BY-DAY setting. Statistically-indistinguishable hyperparameter values for each model type are included in this plot. In this realistic setting, DRBMs are superior to SVMs both in mean performance (statistically significant) and in terms of robustness (lower variance). **Right:** Accuracies achieved by SVMs and DRBMs under the REALISTIC-BY-RANGE setting.

be absorbed by the residuals. Let $\mathsf{Acc}_i$ 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, since this is about them that we are trying to carry out inferences.

The resulting test accuracy for experiment $i$ is represented as

$$\mathsf{Acc}_i = \beta_{h(i)} + \varepsilon_i, \tag{10}$$

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, Searle, and Neuhaus 2008; Pinheiro and 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 $\varepsilon_i$ absorbs all the variability in the accuracy $\mathsf{Acc}_i$ 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 $\varepsilon_{SVM}$ and $\varepsilon_{DRBM}$ 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 $\varepsilon_{SVM}$ and $\varepsilon_{DRBM}$ are both drawn from a normal distribution, the ratio

$$\frac{\mathrm{Var}[\varepsilon_{SVM}]}{\mathrm{Var}[\varepsilon_{DRBM}]}$$

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.[1] 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 (10) to experiment results, and plot the pattern of residuals in Fig. 7 (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 (10) 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`),

[1] 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.

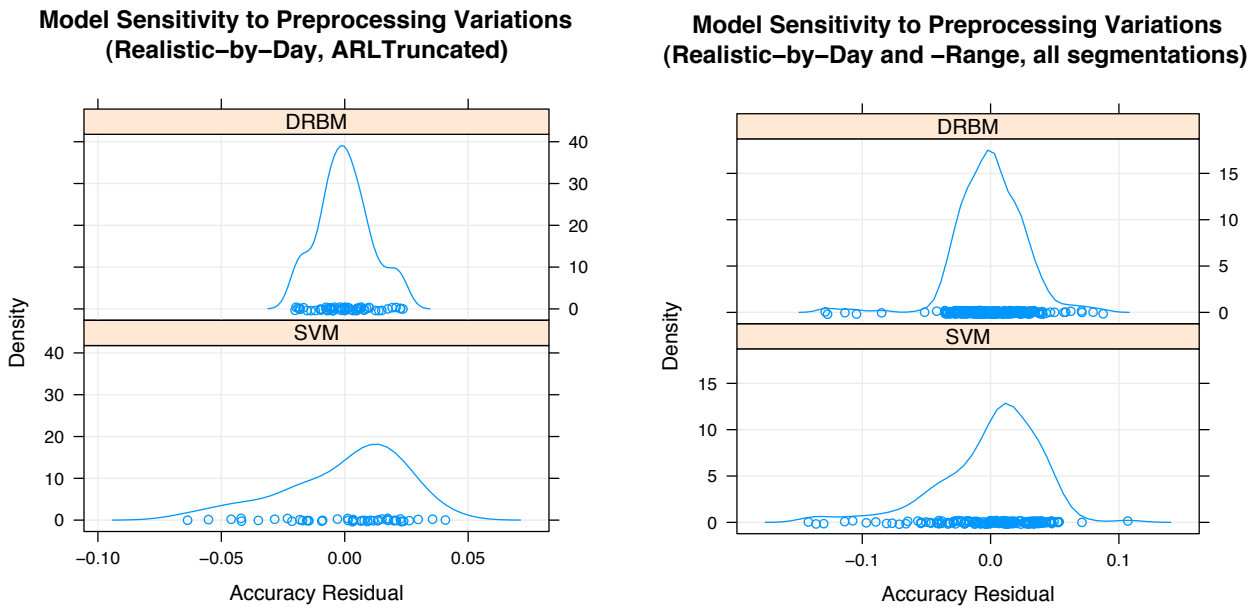


**Figure 7**: **Left:** Sensitivity of SVMs and DRBMs to the choice of preprocessing for the `ARLTruncated` segmentation, under the REALISTIC-BY-DAY setting. **Right:** Same results, across all segmentations and for both REALISTIC-BY-DAY and REALISTIC-BY-RANGE settings. DRBMs are seen to exhibit lower variance than SVMs, implying that they are less sensitive to the choice of preprocessing hyperparameters.

- Evaluation setting (either REALISTIC-BY-DAY or REALISTIC-BY-RANGE),
- and Number of FFT windows.

Although the latter factor is technically part of preprocessing, we have seen (e.g. Fig. 2) 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

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

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 $\varepsilon_i$ 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, Searle, and Neuhaus 2008).

However, analysis of the residuals can proceed as previously. Figure 7 (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.

**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 | $79.1\% \pm 1.4\%$ | $80.8\% \pm 1.4\%$ |

### 5.4 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) and 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).

## 6 Conclusion

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 two types of classifiers: Support Vector Machines (SVMs) and Discriminative Restricted Boltzmann Machines (DRBMs). Although both SVMs and DRBMs exhibit comparable final performance when selecting the best models, we note 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.

## 7 Acknowledgements

This work was supported by the S3 INFORMATION PROCESSING DIRECTORATE (Vince Mirelli) under the auspices of the U.S. Army Research Office Scientific Services Program administered by Battelle (Delivery Order 0258, Contract No. W911NF-07-D-0001).

## 8 References

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning to appear*.

Bengio, Y., P. Lamblin, D. Popovici, and H. Larochelle (2007). Greedy Layer-Wise Training of Deep Networks. In B. SCHÖLKOPF, J. PLATT, and T. HOFFMAN (Eds.), *Advances in Neural Information Processing Systems 19*, pp. 153–160. Cambridge, MA: MIT Press.

Bouchard, G. and B. Triggs (2004, August). The Tradeoff Between Generative and Discriminative Classifiers. In *IASC International Symposium on Computational Statistics (COMPSTAT)*, Prague, pp. 721–728.

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

Carreira-Perpiñán, M. A. and G. Hinton (2005). On Contrastive Divergence Learning. In R. G. COWELL and Z. GHAHRAMANI (Eds.), *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, Jan 6-8, 2005, Savannah Hotel, Barbados*, pp. 33–40. Society for Artificial Intelligence and Statistics. (Available electronically at http://www.gatsby.ucl.ac.uk/aistats/).

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

Croarkin, C. and P. Tobias (Eds.) (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., J. H. Hansen, and J. G. Proakis (1999). *Discrete-Time Processing of Speech Signals*. John Wiley & Sons / IEEE Press.

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 (Eds.), *Computational Neuroscience: Theoretical Insights into Brain Function*. Elsevier.

Hinton, G. E., S. Osindero, and Y. W. Teh (2006). A Fast Learning Algorithm for Deep Belief Networks. *Neural Computation 18*(7), 1527–1554.

Hsu, C.-W. and C.-J. Lin (2002). A Comparison of Methods for Multi-class Support Vector Machines. *IEEE Trans. Neural Networks 13*(5), 415–425.

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

Lasserre, J. A., C. M. Bishop, and T. P. Minka (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*, Washington, DC, USA, pp. 87–94. IEEE Computer Society.

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

Ng, A. Y. and M. I. Jordan (2002). On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes. In T. G. DIETTERICH, S. BECKER, and Z. GHAHRAMANI (Eds.), *Advances in Neural Information Processing Systems 14*, Cambridge, MA, pp. 841–848. MIT Press.

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

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

Ranzato, M., C. Poultney, S. Chopra, and Y. LeCun (2007). Efficient Learning of Sparse Representations with an Energy-Based Model. In B. SCHÖLKOPF, J. PLATT, and T. HOFFMAN (Eds.), *Advances in Neural Information Processing Systems 19*. MIT Press.

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

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

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

Welling, M., M. Rosen-Zvi, and G. Hinton (2005). Exponential Family Harmoniums with an Application to Information Retrieval. In L. K. SAUL, Y. WEISS, and L. BOTTOU (Eds.), *Advances in Neural*

*Information Processing Systems 17*, pp. 1481–1488. Cambridge, MA: MIT Press.