

---

# Super-resolution of Time-series Labels for Bootstrapped Event Detection

---

Ivan Kiskin<sup>\*1</sup> Udeepa Meepegama<sup>\*2</sup> Stephen Roberts<sup>1</sup>

## Abstract

Solving real-world problems, particularly with deep learning, relies on the availability of abundant, quality data. In this paper we develop a novel framework that maximises the utility of time-series datasets that contain only small quantities of expertly-labelled data, larger quantities of weakly (or coarsely) labelled data and a large volume of unlabelled data. This represents scenarios commonly encountered in the real world, such as in crowd-sourcing applications. In our work, we use a nested loop using a Kernel Density Estimator (KDE) to super-resolve the abundant low-quality data labels, thereby enabling effective training of a Convolutional Neural Network (CNN). We demonstrate two key results: a) The KDE is able to super-resolve labels more accurately, and with better calibrated probabilities, than well-established classifiers acting as baselines; b) Our CNN, trained on super-resolved labels from the KDE, achieves an improvement in F1 score of 22.1 % over the next best baseline system in our candidate problem domain.

## 1. Introduction

Finely-labelled data are crucial to the success of supervised and semi-supervised approaches to classification algorithms. In particular, common deep learning approaches (Bishop, 1995; LeCun et al., 2015) typically require a great number of data samples to train effectively (Krizhevsky et al., 2012; Rolnick et al., 2017). In this work, a *sample* refers to a section taken from a larger time-series dataset of audio. Often, these datasets lack large quantities of fine labels as producing them is extremely costly (requiring exact marking of start and stop times). The common distribution of data in these domains is such that there are small quantities of

<sup>\*</sup>Equal contribution <sup>1</sup>Machine Learning Research Group, University of Oxford, Oxford, United Kingdom <sup>2</sup>Department of Physics, University of Oxford, Oxford, United Kingdom. Correspondence to: Ivan Kiskin <ikiskin@robots.ox.ac.uk>.

expertly-labelled (finely) data, large quantities of *weakly* (coarsely) labelled data, and a large volume of unlabelled data. Here, *weak* labels refer to labels that indicate one or more events are present in the sample, although do not contain the information as to the event frequency nor the exact location of occurrence(s) (illustrated in Section 2.2, Fig. 2). Our goal therefore is to improve classification performance in domains with variable quality datasets.

Our key contribution is as follows. We propose a framework that combines the strengths of both traditional algorithms and deep learning methods, to perform multi-resolution Bayesian bootstrapping. We obtain probabilistic labels for pseudo-fine labels, generated from weak labels, which can then be used to train a neural network. For the label refinement from weak to fine we use a Kernel Density Estimator (KDE).

The remainder of the paper is organised as follows. Section 2 discusses the structure of the framework as well as the baseline classifiers we test against. Section 3 describes the datasets we use and the details of the experiments carried out. Section 4 presents the experimental results, while Section 5 concludes.

## 2. Methodology

### 2.1. Framework Overview

Our framework is separated into an inner and outer classifier in cascade as in Figure 1. For the inner classifier we extract features from the finely and weakly-labelled audio data using the two-sample Kolmogorov-Smirnov test for features of a log-mel spectrogram (Section 2.2). We train our inner classifier, the Gaussian KDE (Section 2.3), on the finely-labelled data and predict on the weakly-labelled data.

For the outer classifier we extract the feature vectors from an unlabelled audio dataset using the log-mel spectrogram. We then train our outer classifier, a CNN, (Section 2.4) on the finely-labelled data and the resulting pseudo-finely labelled data output by the Gaussian KDE. The details of our data and problem are found in Section 3. Code will be made available on [https://github.com/HumBug-Mosquito/weak\\_labelling](https://github.com/HumBug-Mosquito/weak_labelling).

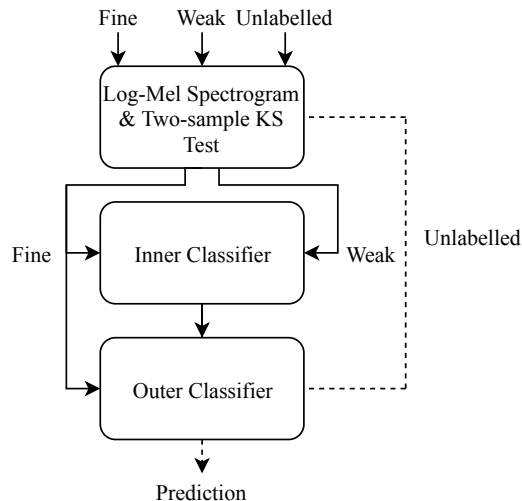


Figure 1. Framework comprising a feature extraction & selection layer, an inner classifier and an outer classifier. The arrows represent data flows.

## 2.2. Feature Extraction and Selection

The CNN uses the log-mel spectrogram (as in Fig. 2) as it has recently become the gold standard in feature representation for audio data (Hayashi et al., 2017; Kong et al., 2019). The input signal is divided into 0.1 second windows and we compute 128 log-mel filterbank features. Thus, for a given 100 seconds of audio input, the feature extraction method produces a  $1000 \times 128$  output.

The two-sample Kolmogorov-Smirnov (KS) test (Ivanov et al., 2012) is a non-parametric test for the equality of continuous, one-dimensional probability distributions that can be used to compare two samples. This measure of similarity is provided by the Kolmogorov-Smirnov statistic which quantifies a distance between the empirical distribution functions of the two samples. We use the KS test to select a subset of the 128 log-mel features, that are maximally different between the two classes to feed into the classifiers. We choose  $N$  features with the largest KS statistics. Fig. 2 illustrates that the process to find maximally different feature pairs, correctly chooses frequencies of interest. For example, if the noise file is concentrated in high frequencies (as in Fig. 2), the KS feature selection process chooses lower harmonics of the training signal (a mosquito flying tone) as features to feed to the algorithms. Conversely, for low-frequency dominated noise, higher audible harmonics of the event signal are identified.

## 2.3. Gaussian Kernel Density Estimation

Kernel density estimation (KDE) or Parzen estimation (Scott, 2015; Parzen et al., 1962) is a non-parametric method for estimating a  $d$ -dimensional probability density function

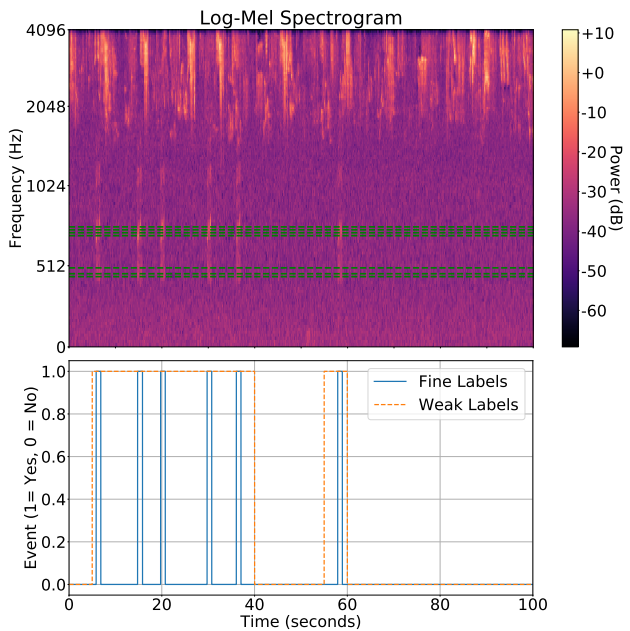


Figure 2. From top to bottom: Log-mel spectrogram of 100 seconds of audio data at a signal-to-noise ratio of  $-15$  dB. The KS-selected features are shown as green dashed lines; The corresponding fine and weak labels for the above log-mel spectrogram.

$f_{\mathbf{X}}(\mathbf{x})$  from a finite sample  $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$ ,  $\mathbf{x}_i \in \mathbb{R}^d$ , by convolving the empirical density function with a kernel function.

We then use Bayes' theorem to calculate the posterior over class 1

$$p(C_1 | \mathbf{x}) = \frac{p(\mathbf{x} | C_1)p(C_1)}{p(\mathbf{x} | C_0)p(C_0) + p(\mathbf{x} | C_1)p(C_1)}, \quad (1)$$

where  $p(\mathbf{x} | C_k)$  is the KDE per class  $C_k$ , with  $C_1$  representing the event class and  $C_0$  the noise class (i.e. non-event).

## 2.4. Convolutional Neural Network

With our scarce data environment we use a CNN and dropout with probability  $p = 0.2$  (Srivastava et al., 2014). Our proposed architecture, given in Fig. 3, consists of an input layer connected sequentially to a single convolutional layer and a fully connected layer. The CNN is trained for 10 epochs with SGD (Bottou et al., 2010), and all activations are ReLUs. We use this particular architecture due to constraints in data size (Kiskin et al., 2018) and therefore have few layers and fewer parameters to learn.

## 2.5. Traditional Classifier Baselines

We compare our inner classifier, the Gaussian KDE, with more traditional classifiers that are widely used in machine learning: Linear Discriminant Analysis (LDA), Gaussian

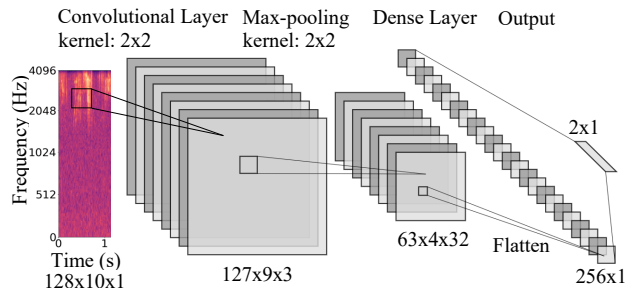


Figure 3. The CNN architecture. Spectrogram of mosquito recording fed as input to convolutional layer with 32 filters and kernel  $2 \times 2 \times 1$ . Generated feature maps are down-sampled by a max-pooling layer from  $127 \times 9$  to  $63 \times 4$ . It is then connected to a dense layer with 256 neurons and finally the output with 2 neurons.

Naïve Bayes (GNB), support vector machines using a radial basis function kernel (RBF-SVM), random forests (RF) and a multilayer perceptron (MLP).

### 3. Experiments

#### 3.1. Description of Datasets

The most common scenario where mixed quality labels can be found is in crowd-sourcing tasks (Cartwright et al., 2019; Deng et al., 2009; Lin et al., 2014), or any challenge where data collection is expensive. The HumBug (Zooniverse, 2019), (Li et al., 2017) project utilises crowd-sourcing, forming the main motivation for this research, as well as the basis for our signal<sup>1</sup>. The event signal consists of a stationary real mosquito audio recording with a duration of 1 second. The noise file is a non-stationary section of ambient forest sound. The event signal is placed randomly throughout the noise file at varying signal-to-noise ratios (SNRs), to create quantifiable data for prototyping algorithms. There is a class imbalance of 1 second of event to 9 seconds of noise in the finely-labelled data and this is propagated to the weakly-labelled and unlabelled datasets. We include 100 seconds of expert, finely-labelled data, 1000 seconds of weakly-labelled data, and a further 1000 seconds of unlabelled test data. To report performance metrics, we create synthetic labels at a resolution of 0.1 seconds for the finely-labelled data, and on the order of 5 seconds for the weakly-labelled data. We choose 5 seconds as to allow the labeller to have temporal context when classifying audio as an event or non-event. As the listener is presented randomly sampled (or actively sampled (Houlsby et al., 2011; Naghshvar et al., 2012)) sections of audio data, a section much shorter than 5 seconds would make the task of tuning

<sup>1</sup>The overall goal of HumBug is real-time mosquito species recognition to identify potential malaria vectors in order to deliver intervention solutions effectively.

into each new example very difficult due to the absence of a reference signal.

#### 3.2. Experimental Design

We evaluate our inner model against the baseline classifiers with two experiments and finally test the overall performance of the framework utilising the outputs of the various inner classifiers. We make the assumption that the accuracy of the weak labels is 100%. Therefore, all the classifiers predict over the coarse class 1 labelled data only. Additionally, the priors we use in Eq. 1 for our Gaussian KDE model are set such that  $p(C_0) = p(C_1) = 0.5$ . This is to reflect that, since the audio sample is weakly labelled 1, each data point is equally likely to be in fine class 0 or 1.

Generative models, such as the Gaussian KDE obtain a performance boost from the additional information provided by the coarse class 0 data as this allows it to better model the class 0 distribution. Conversely, discriminative models such as the SVM, RF and MLP take a hit in performance because the decision boundary that they create over-fits to the class 0 data points due to the increased class imbalance. We therefore train the LDA, GNB, SVM, RF and MLP on the finely-labelled data only, whereas the Gaussian KDE is trained on both the finely-labelled data and the coarse class 0 data.

### 4. Classification Performance

For each SNR we run 40 iterations, varying the time location of the injected signals, as well as the random seed of the algorithms. After applying median filtering, with a filter window of 500 ms, we see the results in Fig. 4. The F1-score gradually increases as expected from the threshold of detection to more audible SNRs. The decay of performance at the lower SNRs can be partially accounted for by the two-sample KS test used for feature selection failing to choose features of interest due to the increased noise floor. We observe a significant benefit to using the Gaussian KDE, which when combined with temporal averaging helps recover the dynamic nature of the signal (namely that there is correlation between neighbouring values in the time-series).

Fig. 4 shows that the Gaussian KDE predicts better calibrated probabilities than the other baseline classifiers. This is shown by applying rejection (Bishop, 1995; Hanczar and Dougherty, 2008) in addition to the median filtering. The rejection window for the output probabilities is  $0.1 < p(C_1|\mathbf{x}) < 0.9$ . The Gaussian KDE improves significantly in performance, especially at the lower SNRs; however it should be noted that the F1-score is evaluated on the remaining data after rejection. The Gaussian KDE rejects a large proportion of the data at lower SNRs, showing

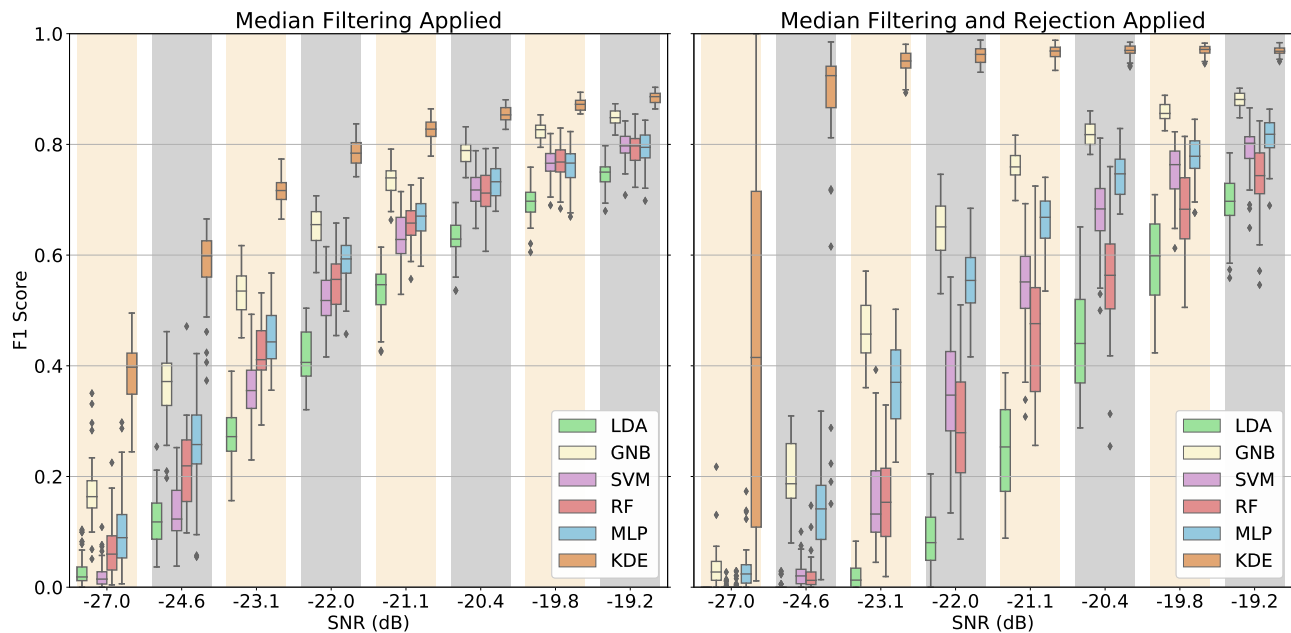


Figure 4. From left to right: Boxplots showing results of first two experiments, grouped by SNR. LDA, SVM, RF and MLP are trained on finely-labelled data only whilst the Gaussian KDE is trained on the finely-labelled data and coarse class 0 data.

that the probabilities are at either extremes only when the model is confident in its predictions.

The final experiment tests the overall framework with input to a CNN from pseudo-finely labelled data with median filtering and rejection applied. Table 1 shows that using the framework in conjunction with any of the inner classifiers outputs outperforms a regular CNN trained on the coarse data. Furthermore, training the CNN on the output of the Gaussian KDE significantly improves detection of events by 22.1% over the best baseline system, the CNN(GNB). We also show that using the strongest inner classifier (KDE) alone results in vastly lower precision and recall scores to the bootstrapping approach advocated here, which sees an improvement of 0.22 to the F1-score gained by incorporating the CNN into the pipeline with the KDE.

Table 1. CNN outer classifier: Metrics reported as means  $\pm$  one standard deviation at an SNR of  $-19.8$  dB for 40 iterations

Classifier	F1-score	Precision	Recall
<b>CNN(KDE)</b>	<b><math>0.729 \pm 0.034</math></b>	<b><math>0.719 \pm 0.029</math></b>	<b><math>0.744 \pm 0.031</math></b>
CNN(MLP)	$0.435 \pm 0.022$	$0.667 \pm 0.026$	$0.322 \pm 0.029$
CNN(RF)	$0.320 \pm 0.031$	$0.419 \pm 0.035$	$0.259 \pm 0.034$
CNN(SVM)	$0.338 \pm 0.024$	$0.484 \pm 0.024$	$0.259 \pm 0.022$
CNN(GNB)	$0.597 \pm 0.023$	$0.654 \pm 0.028$	$0.549 \pm 0.023$
CNN(LDA)	$0.307 \pm 0.027$	$0.571 \pm 0.026$	$0.210 \pm 0.023$
CNN(Coarse)	$0.174 \pm 0.036$	$0.095 \pm 0.031$	$0.923 \pm 0.039$
KDE	$0.506 \pm 0.021$	$0.518 \pm 0.021$	$0.502 \pm 0.024$

## 5. Conclusions & Further Work

### 5.1. Conclusions

This paper proposes a novel framework utilising a Gaussian KDE for super-resolving weakly-labelled data to be fed into a CNN to predict over unlabelled data. Our framework is evaluated on synthetic data and achieves an improvement of 22.1% in F1-score over the best baseline system. We thus highlight the value label super-resolution provides in domains with only small quantities of finely-labelled data, a problem in the literature that is only sparsely addressed to date.

### 5.2. Further Work

To leverage the probabilistic labels outputted by the inner classifier, a suitable candidate for the outer classifier is a loss-calibrated Bayesian neural network (LC-BNN). This combines the benefits of deep learning with principled Bayesian probability theory (Cobb et al., 2018).

Due to computational limitations, optimisation of the hyper-parameters was infeasible. Future work plans to use Bayesian Optimisation (Snoek et al., 2012) for this tuning.

Finally, following the promising results of this paper, the next step is application to real datasets.

## References

- Bishop, C.M., (1995). Neural networks for pattern recognition. Oxford University Press.
- Bottou, L., (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177-186.
- Cartwright, M., Dove, G., Méndez, A.E.M. and Bello, J.P., (2019). Crowdsourcing multi-label audio annotation tasks with citizen scientists. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 4-9.
- Cobb, A.D., Roberts, S.J. and Gal, Y., (2018). Loss-calibrated approximate inference in Bayesian neural networks. In *arXiv preprint arXiv:1805.03901*
- Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248-255.
- Hanczar, B. and Dougherty, E.R., (2008). Classification with reject option in gene expression data. In *Bioinformatics*, 24(17):1889-1895.
- Hayashi, T., Watanabe, S., Toda, T., Hori, T., Le Roux, J. and Takeda, K., (2017). BLSTM-HMM hybrid system combined with sound activity detection network for polyphonic sound event detection. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 766-770.
- Houlsby, N., Huszár, F., Ghahramani, Z. and Lengyel, M., (2011). Bayesian active learning for classification and preference learning. In *arXiv preprint arXiv:1112.5745*.
- Ivanov, A. and Riccardi, G., (2012). Kolmogorov-Smirnov test for feature selection in emotion recognition from speech. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5125-5128
- Kiskin, I., Zilli, D., Li, Y., Sinka, M., Willis, K. and Roberts, S., (2018). Bioacoustic detection with wavelet-conditioned convolutional neural networks. In *Neural Computing and Applications*, pp.1-13
- Kong, Q., Xu, Y., Sobieraj, I., Wang, W. and Plumbley, M.D., (2019). Sound Event Detection and Time-Frequency Segmentation from Weakly Labelled Data. In *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 27(4):777-787.
- Krizhevsky, A., Sutskever, I. and Hinton, G.E., (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097-1105.
- LeCun, Y., Bengio, Y. and Hinton, G., (2015). Deep learning. In *Nature*, 521(7553):436.
- Li, Y., Zilli, D., Chan, H., Kiskin, I., Sinka, M., Roberts, S. and Willis, K., (2017). Mosquito detection with low-cost smartphones: data acquisition for malaria research. In *arXiv preprint arXiv:1711.06346*.
- Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740-755.
- Naghshvar, M., Javidi, T. and Chaudhuri, K., (2012). Noisy bayesian active learning. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1626-1633
- Parzen, E., (1962). On estimation of a probability density function and mode. In *The annals of mathematical statistics*, 33(3):1065-1076.
- Rolnick, D., Veit, A., Belongie, S. and Shavit, N., (2017). Deep learning is robust to massive label noise. In *arXiv preprint arXiv:1705.10694*.
- Scott, D.W., (2015). Multivariate density estimation: theory, practice, and visualization. John Wiley & Sons.
- Snoek, J., Larochelle, H. and Adams, R.P., (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pp. 2951-2959
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., (2014). Dropout: a simple way to prevent neural networks from overfitting. In *The Journal of Machine Learning Research*, 15(1):1929-1958.
- Zooniverse Humbug page (2019), <https://www.zooniverse.org/projects/yli/humbug>. Accessed: 2019-04-29