Shared nearest neighbors match kernel for bird songs identification - LifeCLEF 2015 challenge

Alexis Joly[1], Valentin Leveau[1,3], Julien Champ[2], and Olivier Buisson[3]

[1] Inria, LIRMM, Montpellier, France
alexis.joly@inria.fr valentin.leveau@inria.fr
[2] Inra, AMAP, LIRMM, Montpellier, France
julien.champ@inra.fr
[3] Institut National de l'Audiovisuel (INA), Bry-sur-Marne, France
olivier.buisson@ina.fr

Abstract. This paper presents a new fine-grained audio classification technique designed and experimented in the context of the LifeCLEF 2015 bird species identification challenge. Inspired by recent works on fine-grained image classification, we introduce a new match kernel based on the shared nearest neighbors of the low level audio features extracted at the frame level. To make such strategy scalable to the tens of millions of MFCC features extracted from the tens of thousands audio recordings of the training set, we used high-dimensional hashing techniques coupled with an efficient approximate nearest neighbors search algorithm with controlled quality. Further improvements are obtained by (i) using a sliding window for the temporal pooling of the raw matches (ii) weighting each low level feature according to the semantic coherence of its nearest neighbors. Results show the effectiveness of the proposed technique which ranked 2nd among the 7 research groups participating to the LifeCLEF bird challenge.

1 Introduction

Building accurate knowledge of the identity, the geographic distribution and the evolution of living species is essential for a sustainable development of humanity as well as for biodiversity conservation. In this context, using multimedia identification tools is considered as one of the most promising solution to help bridging the taxonomic, i.e. the difficulty for common people to name observed living organisms and then produce or access to useful knowledge. The LifeCLEF [10] lab proposes to evaluate this challenge in the continuity of the image-based plant identification task was run within ImageCLEF the years before but with a broader scope (considering birds and fish in addition to plants and audio and video contents in addition to images). This paper particularly reports the participation of Inria ZENITH research group to the audio-based bird identification task. Inspired by some recent works on fine-grained image classification [12], we introduce a new match kernel based on the shared nearest neighbors of the low level audio features extracted at the frame level. Section 2 describes the preliminary audio processing and features extraction steps. Section 3 then presents our new match kernel and the resulting explicit representations to be further classified thanks to a linear supervised classifier (section 4). Section 5 and **??** finally reports and discuss the results we obtained within the LifeCLEF 2015 challenge .

2 Pre-processing and features extraction

The dataset used for this challenge is composed of 33,203 audio recordings belonging to 999 bird species from Brazil area. As various recording devices are used, and because it is difficult to capture these sounds as birds are often far away from the recording devices, many recordings contains a lot of noise. To overcome this problem, we used SoX, the "Swiss Army knife of sound processing programs"[4]. As a first step, we used the *noisered specialised filter*, to filter out noise from the audio, and then we reduce the length of large (i.e. $> 0.1s$) silent passages from audio files to $0.1s$. In order to obtain audio files with ideally no more noise but still enough signal, we tried removing as much noise as possible (using the noisered amount parameter) while guaranteeing that the resulting audio file was at least 20% the size of the initial audio record. After this pre-processing step, we used an open source software framework, marsyas[5], to extract MFCC features with parameters based on the provided audio features in the Birdclef task : MFCC are computed on windows of 11.6 ms, each 3.9 ms, and we additionally derive their speed resulting in 26-dimensional feature vectors (13+13) for each frame.

3 Shared Nearest Neighbors Match Kernel

We consider two recordings I_x and I_y represented by sets of 26-dimensional MFCC features $X = \{\mathbf{x}\}$ and $Y = \{\mathbf{y}\}$. We then build on the *normalized sum match kernel* proposed by [13] to compare feature sets:

$$K(X,Y) = \Phi(X)^T \Phi(Y) = \frac{1}{|X|\,|Y|} \sum_{\mathbf{x}} \sum_{\mathbf{y}} k(\mathbf{x}, \mathbf{y}) \tag{1}$$

where $k()$ is itself a Mercer kernel allowing to compare two individual local features $\mathbf{x}$ and $\mathbf{y}$. In our case, $k()$ is however not defined as a direct matching between $\mathbf{x}$ and $\mathbf{y}$ but rather as the degree of correlation of their matches in a large training set. Let denote as $\mathcal{Z}$ such a training set composed of N 26-dimensional MFCC feature vectors $\mathbf{z}$. We introduce the following *shared nearest neighbors (SNN) match kernel* :

$$K_S(X,Y) = \frac{1}{|X|\,|Y|} \sum_{\mathbf{x}} \sum_{\mathbf{y}} \sum_{\mathbf{z}} \varphi_{\mathbf{x}}(\mathbf{z}).\varphi_{\mathbf{y}}(\mathbf{z}) \tag{2}$$

[4] http://sox.sourceforge.net/

[5] http://marsyas.info/

with $\varphi_{\mathbf{x}}(\mathbf{z})$ a rank-based activation function given by :

$$\varphi_{\mathbf{x}}(\mathbf{z}) = \sqrt{\frac{log(K) - log(r_{\mathbf{x}}(\mathbf{z}))}{log(K)}} \tag{3}$$

where $r_{\mathbf{x}}(\mathbf{z}) : \mathcal{Z} \to R^{+}$ is a ranking function returning the rank of an item $\mathbf{z} \in \mathcal{Z}$ according to its distance to $\mathbf{x}$ and K is the maximum number of items returned by this ranking function. The distance itself could be a L_2 metric in the original feature space but, as we will see in section 3, we use in practice a more efficient Hamming embedding scheme. Whatever the distance used, the intuition of the *SNN match kernel* is that it counts the number of common neighbors in the neighborhood of $\mathbf{x}$ and in the one of $\mathbf{y}$. The product $k_{\mathbf{z}}(\mathbf{x}, \mathbf{y}) = \varphi_{\mathbf{x}}(\mathbf{z}).\varphi_{\mathbf{y}}(\mathbf{z})$ is actually equal to one if $\mathbf{z}$ is the nearest neighbor of both $\mathbf{x}$ and $\mathbf{y}$ and close to zero if $\mathbf{z}$ is not in the top neighbors of either $\mathbf{x}$ or $\mathbf{y}$.
Using this *shared nearest neighbors kernel* instead of a more classical distance in the feature space has several justifications and advantages. First, shared-neighbors techniques are known to overcome several shortcomings of traditional metrics. They are notably less sensitive to the dimensionality curse, more robust to noisy data and more stable over unusual features distribution [2, 5] . Measuring the similarity between features by the degree to which their neighbourhoods in the training set resemble one another is actually a form of generative metric learning. Features belonging to dense clusters are actually more likely to share neighbors than uniformly distributed and isolated features. So that their contribution in the global kernel will be enhanced. Secondly, using an indirect matching rather than a direct one allows to have en explicit formulation of the embedded space $\Phi(X)$. By factorizing equation 3, it is actually easy to show that $K_S(X, Y) = \Phi_S(X)^T \Phi_S(Y)$ with:

$$\Phi_S(X) = \sum_{\mathbf{i}=1}^{N} \frac{1}{|X|} \sum_{\mathbf{x}} \varphi_{\mathbf{x}}(\mathbf{z}_i).\overrightarrow{e_i} \tag{4}$$

So that, the explicit N-dimensional feature vector $\Phi(X)$ representing each audio recording in the training set can be computed before training a simple linear classifier on top of them. This principle of this approach was already introduced in the *intermediate matching kernel* of [1] and further re-used in many methods including the NBNN kernel of [15]. Such methods did however rely on the distance between the features of the candidate object X and the ones of the training set $\mathcal{Z}$ so that they did not benefit from the nice properties of the SNN-kernel. Last but not least, one of the main advantage of the *SNN match kernel* is that is that it can be easily converted to a sparse representation as the ratio of the number of values close to zeros is very high. Only the features $\mathbf{z}$ lying in the top neighbors of both $\mathbf{x}$ and $\mathbf{y}$ lead to consistent component values. In practice, it is therefore sufficient to consider only the top-m neighbors of each feature $\mathbf{x}$ and $\mathbf{y}$ to get a good approximation of $K(X, Y)$. This allows using efficient nearest neighbors search techniques to construct the explicit representations $\Phi(X)$ and to use an efficient sparse encoding when training linear classifiers on top of them.

Temporal pooling of the raw SNN-based representations As elegant as the explicit representations $\Phi_S(X)$ is, it does not conduct to good classification results in practice. It's very high dimensionality, equals to the number of features in the training set (often millions), actually leads to strong overfitting even when using $L2$ regularizers with high values of the regularization control parameter λ. It is therefore required to group the individual matches of the SNN kernel before deriving an effective explicit representation. In this work, we do focus on the *temporal* pooling of the raw matches rather than aggregating them in the feature space as done in many popular image representations such as BoW, Fisher Vectors or VLAD. We consequently loose some generalization capacity in the feature space compared to these methods but on the other side we strongly boost the locality, the interpretability and the discrimination of the trained audio patterns.
Practically, our temporal pooling algorithm first aggregates the raw matches within a sliding window (centered around each frame) and then keep the max score over the whole record. More formally, we can reformulate our explicit formulation of Equation 4 as:

$$\Phi^w_S(X) = \sum_{\mathbf{m}=1}^{M} \left(\max_{t_i \in [1,T_m]} \sum_{\mathbf{t}=t_i-(w/2)}^{t_i+(w/2)} \sum_{\mathbf{x}\in X} \varphi_{\mathbf{x}}(\mathbf{z}^m_t) \right) .\overrightarrow{e_m} \tag{5}$$

where M is the number of audio recordings in the training set, T_m the number of frames of the m-th recording and $\mathbf{z}^m_t$ the MFCC feature of the t-th frame of the m-th recording. The size w of the sliding window was trained by cross-validation and then fixed to $w = 1000$ frames (resulting in a sliding window of 3.9 seconds).

Approximate K-NN search scheme In practice, to speed up the computation of our SNN based representations, the ranking function $r_{\mathbf{x}}(\mathbf{z}) : \mathcal{Z} \rightarrow R^+$ is implemented as an approximate nearest neighbors search algorithm based on hashing and probabilistic accesses in the hash table. It takes as input each query feature $\mathbf{x}$ of the audio recording I_x to be described and returns a set of K approximated neighbors in $\mathcal{Z}$ with an approximated rank $r'_{\mathbf{x}}(\mathbf{z})$. The exact ranking function $r_{\mathbf{x}}(\mathbf{z})$ is simply replaced by this approximated ranking function in all equations above. Note that the features $\mathbf{z} \in \mathcal{Z}$ that are not returned in the top-K approximated nearest neighbors are simply *removed* from the SNN match kernels equations conducting to a considerable reduction of the computation time. Consequently, they are implicitly considered as having a rank-based activation function $\varphi_{\mathbf{x}}(\mathbf{z})$ equal to zero which is a good approximation as their rank is supposed to be higher than K.

Let us now describe more precisely our approximate nearest neighbors indexing and search method. It first compresses the original feature vectors $\mathbf{z} \in \mathcal{Z}$ into compact binary hash codes $\mathbf{h}(\mathbf{z})$ of length b. This is done by using RMMH [8], a recent data-dependent hash function family, in order to embed the original feature vectors in compact binary hash codes of $b = 128$ bits (the parameter M of RMMH was fixed to $M = 32$). The distance between any two features $\mathbf{x}$ and $\mathbf{z}$ can then be efficiently approximated by the Hamming distance between their respective 128-length hash codes $\mathbf{h}(\mathbf{z})$ and $\mathbf{h}(\mathbf{x})$. According to our experiments, this hashing method provides in our context better performances than several other tested methods, including random projections or hamming embedding [6] (orthogonal random projections).

To avoid scanning the whole dataset, the hash codes $\mathbf{h}(\mathbf{z})$ derived from the local features of the training set $\mathcal{Z}$ are then indexed in a hash table whose keys are the t-length prefix of the hash codes $\mathbf{h}(\mathbf{z})$. At search time, the hash code $\mathbf{h}(\mathbf{x})$ of a query feature $\mathbf{x}$ is computed as well as its t-length prefix. We then use a probabilistic multi-probe search algorithm inspired by the one of [7] to select the buckets of the hash table that are the most likely to contain exact nearest neighbors. This is done by using a probabilistic search model that is trained offline on the exact m-nearest neighbors of M sampled features $\mathbf{z} \in \mathcal{Z}$. We however use a simpler search model than the one of [7]. We actually use a normal distribution with independent components parameterized by a single vector σ that is trained over the exact nearest neighbors of the training samples. At search time, we also use a slightly different probabilistic multi-probe algorithm trading stability for time. Instead of probing the buckets by decreasing probabilities, we rather use a greedy algorithm that computes the probability of neighboring buckets and select only the ones having a probability greater than a threshold ζ that is fixed over all queries. The value of ζ is trained offline on M training samples and their exact nearest neighbors so as to reach on average cumulative probability α over the visited buckets. In our experiments, we always used $\alpha = 0.80$ meaning that on average we retrieve 80% of the exact nearest neighbors in the original feature space. Once the most probable buckets have been selected, the refinement step computes the Hamming distance between $\mathbf{h}(\mathbf{x})$ and the $\mathbf{h}(\mathbf{z})$'s belonging to the selected buckets and keep only the top-m matches thanks to a max heap.

Weak semantic weighting As we are in the case of weakly annotated audio recordings with multiple classes (primary and secondary species) and highly cluttered contexts, we suggest improving our SNN match kernel by weighting the query features according to the semantic coherence of their k nearest neighbours. We therefore compute a discrimination score $f(\mathbf{x})$ for all MFCC features $\mathbf{x} \in X$ of a given audio recording I_X. A weak label $l(\mathbf{x})$ is first estimated for each $\mathbf{x}$ as the most represented label within the k-nearest neighbors of $\mathbf{x}$ in the training set (actually the ones computed by the hash-based k-nn search method described in section 3). The semantic weight $f(\mathbf{x})$ is then computed as the percentage of the k-nearest neighbors having the same weak label than the feature itself (i.e. the percentage of k-nearest neighbors whose label is equal to $l(\mathbf{x})$). Finally, our representation of a given audio recording I_X becomes:

$$\varPhi_S^{w}{}'(X) = \sum_{\mathbf{m}=1}^{M} \left(\max_{t_i \in [1,T_m]} \sum_{\mathbf{t}=t_i-(w/2)}^{t_i+(w/2)} \sum_{\mathbf{x}\in X} f(x).\varphi_{\mathbf{x}}(\mathbf{z}_t^m) \right) .\overrightarrow{e_m} \tag{6}$$

4 Training and classification

To achieve an effective supervised classification task, we trained a linear discriminant model on top of our proposed SNN matching-based representations (cf. Equation 5). This requires first building the representations of all audio recordings in the training set and then in learning as many linear classifiers as the number of species in the training set. The resulting linear classifiers are of the form:

$$h(\varPhi_S^{w}{}'(X)) = \omega^T.\varPhi_S^{w}{}'(X) + b$$

so that they interestingly affect weights ω_j to each audio recording in the training set according to its relevance for the targeted class (rather than affecting weights to the individual MFCC features as in the raw representation of Equation 4). In our experiments, we used a linear support vector machine for training these discriminant linear models. We more precisely used the LibLinear implementation of the scikit-learn library with a squared hinge loss function and a L_2 penalty. The C parameter of the SVM was fixed to $C = 100.0 * weight(class)$ where $weight(class)$ is a class-dependent weight that is automatically adjusted to be inversely proportional to the class frequency. Finally, the scores returned by the SVM are converted into probabilities using the following p-value test:

$$P(class) = \frac{1}{2}\left(1 + erf\left(\frac{1}{\sqrt(2)}\frac{s(class) - \mu(class)}{\sigma(class)}\right)\right)$$

where erf is the Gauss error function and $\mu(class)$ and $\sigma(class)$ are respectively the mean and the standard deviation of the SVM score across the considered class. We will see in the experiments that this conversion provides a noticeable accuracy improvement.

5 Experiments and results

5.1 Dataset and task

The LifeCLEF 2015 bird dataset [3] is built from the Xeno-canto collaborative database [6] involving at the time of writing more than 140k audio records covering 8700 bird species observed all around the world thanks to the active work of more than 1400 contributors. The subset of Xeno-canto data used for the 2015 edition

[6] http://www.xeno-canto.org/

of the task contains 33,203 audio recordings belonging to 999 bird species in Brazil area, i.e. the ones having the more recordings in Xeno-canto database. The dataset contains minimally 14 recordings per species and minimally 10 different recordists per species.Audio records are associated to various metadata such as the type of sound (call, song, alarm, flight, etc.), the date and localization of the observations (from which rich statistics on species distribution can be derived), some textual comments of the authors, multilingual common names and collaborative quality ratings (more details can be found in [3]). The task was evaluated as a bird species retrieval task. A part of the collection was delivered as a training set available a couple of months before the remaining data is delivered. The goal was to retrieve the singing species among the top-k returned for each of the undetermined observation of the test set. Participants were allowed to use any of the provided metadata complementary to the audio content but we did not in our own submissions.

Run Name	MAP 2(without Background Species)	MAP 2 (with Background Species)
MNB TSA Run 4	0.454	0.414
MNB TSA Run 3	0.442	0.411
MNB TSA Run 2	0.442	0.405
MNB TSA Run 1	0.424	0.388
INRIA ZENITH Run 2	**0.334**	**0.291**
QMUL Run 1	0.302	0.262
INRIA ZENITH Run 3	**0.292**	**0.259**
INRIA ZENITH Run 1	**0.265**	**0.240**
GOLEM Run 2	0.171	0.149
GOLEM Run 1	0.161	0.139
CHIN. AC. SC. Run 1	0.01	0.009
CHIN. AC. SC. Run 3	0.009	0.01
CHIN. AC. SC. Run 2	0.007	0.008
MARF Run 1	0.006	0.005
MARF Run 2	0.003	0.002
MARF Run 3	0.005	0.005
MARF Run 4	0.000	0.000

Table 1. Official results of LifeCLEF 2015 Bird Task - Our runs are referred as **INRIA Zenith Run 1**, **INRIA Zenith Run 2** and **INRIA Zenith Run 3**

5.2 Submitted runs and results

We submitted three *runs* to be evaluated within the LifeCLEF 2015 challenge:

INRIA Zenith Run 1: This run was not based on the method described in this paper, but on our former instance-based classification method [9] evaluated within the 2014 BirdCLEF challenge [4]. This allows us measuring progresses between that former approach and the new one proposed in this paper. It basically relied on a very similar matching process than the one described in this paper but it did not train any supervised classifier on top of the resulting matching score. It actually only computed the top-30 most similar training records of each query and then used a simple vote on the labels of the retrieve records as classifier. It however included a pre-filtering of the training set that removed the less discriminant MFCC features from the training set.

INRIA Zenith Run 2: The new approach described in this paper.

INRIA Zenith Run 3: The same approach than Run 2 (i.e. the main contribution of that paper), but without the conversion of the SVM scores into probabilities (see section 4).

The results of the whole challenge, including our own results as well as the results of the other participating research groups, are reported in Figure 1 and Table 1.

5.3 Discussion and perspectives

Our system globally achieved very good performance and ranked as the second best one among the 7 participating research groups. Our best run, i.e. the one based on the method proposed in this paper, achieved a mAP of $0,334$ when considering only the primary species of each test recording. This is 3 points better than the state-of-the-art approach of the QMUL research group which makes use of unsupervised feature learning as described in [14] whereas we used classical MFCC features. Also, compared to the mAP of our first run (equals to 0.265), it shows that training discriminant models using our SNN match kernel is much more effective than using our former semantic pruning and instance-based classification approach. The weights learned by the SVM on the pooled matches actually compensate most of the bias involved by the heterogeneity of the noise level in the recordings and the heterogeneity of the recordings length. The intermediate performance of INRIA Zenith Run 3 shows, however, that the conversion of the SVM scores into probabilities plays an important role in the performance of Run2. Our interpretation of this phenomenon is related to the fact that the number of training records per species follows an heavily tailed distribution (as in most biodiversity data). The SVM scores are consequently boosted for the most populated species to the detriment of the less populated ones. Our p-value normalization allows compensating this bias by normalizing the distribution across all classes.

Now, the performance of our approach is still much lower than the best performing system of MNB TSA which has a mAP equal to 0.453. Note that their approach is in essence not so far from ours as they also represent the audio recordings thanks to their matching score in a reference set of audio segments [11]. A major difference however is that they pre-compute a clean set of relevant audio segments whereas we use all the recordings of the training set as vocabulary. They notably consider only the audio recordings with the highest user ratings in the metadata, and, then extract only the segments that are likely to contain a bird song (thanks to bandwidth considerations). A second difference is that their matching is computed at the signal level whereas we are using MFCC features that might loose some important information. We believe that integrating these two additional paradigms within our framework could make it competitive with their approach. Investigating more in depth the semantic pruning strategy that we introduced in [9] but in the context of our new SNN match kernel might for instance be an effective way of further improving the quality of the reference set.

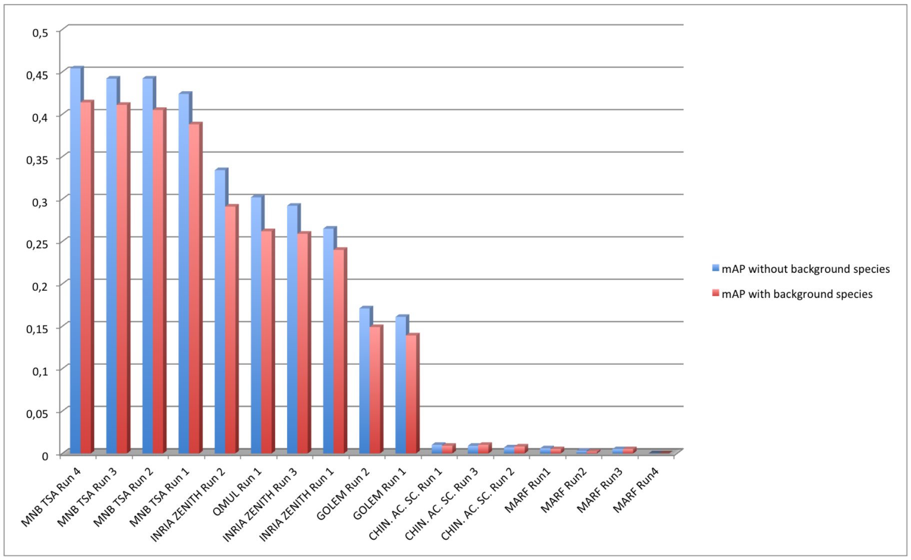

Fig. 1. Official results of LifeCLEF 2015 Bird Task - Our runs are referred as **INRIA Zenith Run 1**, **INRIA Zenith Run 2** and **INRIA Zenith Run 3**

References

1. Boughorbel, S., Tarel, J.P., Boujemaa, N.: The intermediate matching kernel for image local features. In: Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on. vol. 2, pp. 889–894. IEEE (2005)
2. Ertoz, L., Steinbach, M., Kumar, V.: A new shared nearest neighbor clustering algorithm and its applications. In: Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining. pp. 105–115 (2002)
3. Glotin, H., Joly, A., Goëau, H., Vellinga, W.P., Rauber, A.: Lifeclef bird identification task 2015. In: CLEF working notes 2015 (2015)
4. Goëau, H., Glotin, H., Vellinga, W.P., Rauber, A.: Lifeclef bird identification task 2014
5. Jarvis, R.A., Patrick, E.A.: Clustering using a similarity measure based on shared near neighbors. Computers, IEEE Transactions on 100(11), 1025–1034 (1973)
6. Jegou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. Computer Vision–ECCV 2008 pp. 304–317 (2008)
7. Joly, A., Buisson, O.: A posteriori multi-probe locality sensitive hashing. In: Proceedings of the 16th ACM international conference on Multimedia. pp. 209–218. ACM (2008)
8. Joly, A., Buisson, O.: Random maximum margin hashing. In: CVPR. IEEE, Colorado springs, United States (Jun 2011)
9. Joly, A., Champ, J., Buisson, O.: Instance-based bird species identication with undiscriminant features pruning-lifeclef 2014. In: CLEF2014 (2014)
10. Joly, A., Müller, H., Goëau, H., Glotin, H., Spampinato, C., Rauber, A., Bonnet, P., Vellinga, W.P., Fisher, B.: Lifeclef 2014: multimedia life species identification challenges
11. Lasseck, M.: Large-scale identification of birds in audio recordings. In: Working notes of CLEF 2014 conference (2014)
12. Leveau, V., Joly, A., Buisson, O., Valduriez, P.: Kernelizing spatially consistent visual matches for fine-grained classification. In: ICMR 2015 (2015)
13. Lyu, S.: Mercer kernels for object recognition with local features. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. vol. 2, pp. 223–229. IEEE (2005)
14. Stowell, D., Plumbley, M.D.: Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning. PeerJ 2, e488 (2014)
15. Tuytelaars, T., Fritz, M., Saenko, K., Darrell, T.: The nbnn kernel. In: Computer Vision (ICCV), 2011 IEEE International Conference on. pp. 1824–1831. IEEE (2011)