On the Importance of Scalability: Should *k*-NN Genre Classification be Revisited?

Haukur Pálmason, Björn Þór Jónsson, Laurent Amsaleg

School of Computer Science
Reykjavík University
March 2012

TECHNICAL REPORT / RUTR-CS10001

ISSN 1670-5777

On the Importance of Scalability: Should k-NN Genre Classification be Revisited?

Haukur Pálmason[*], Björn Þór Jónsson[*], Laurent Amsaleg[†]

Technical Report RUTR-CS10001, March 2012

Abstract: Personal or professional music collections grow at a very fast pace due to the ease with which digital music can be created, shared, stored and manipulated. As a result, modern music management and browsing require the use of searchable metadata built on low-level features describing the contents of songs, in addition to titles, artists or album names. Musical genre is an example of such metadata, which has been shown to be useful. Currently, the best approaches for automatic genre classification achieve high quality results, with around 91% classification accuracy. These approaches, however, use sophisticated classification algorithms (that are hence quite slow) evaluated on standard music collections (that are small). Using such approaches in the real world, where collections are orders of magnitude larger, is therefore difficult in practice.

In contrast, this report addresses large scale genre classification based on a k-nearest neighbor paradigm. The rationale for using such a simple approach is that it is much faster than all other state-of-the-art schemes, which is required to cope with scale. In the past, the quality of k-NN classification has been deemed insufficient in the literature. We claim, however, that the ability of k-NN schemes to gracefully cope with scale is a strong motivation for working to improve their accuracy. As a first step in this direction, this report shows, through a number of experiments, that k-NN classification can indeed yield quite good results for automatic genre classification. Furthermore, the classification times that we report show that k-NN classification is very fast, which is necessary for real life collections.

Keywords: Music classification; Scalability; k-NN classification.

This work was partially supported by an Icelandic Student Research Fund Grant and by a Reykjavik University Development Fund grant.

[*] Reykjavik University, Menntavegi 1, IS-101 Reykjavík, Iceland. bjorn@ru.is
[†] IRISA–CNRS, Campus de Beaulieu, 35042 Rennes, France. laurent.amsaleg@irisa.fr

Um mikilvægi skalanleika:
Verðskuldar *k*-NN flokkun frekari skoðun?

Haukur Pálmason, Björn Þór Jónsson, Laurent Amsaleg

Tækniskýrsla RUTR-CS10001, Mars 2012

Útdráttur: Stafræn tónlistarsöfn fara ört vaxandi vegna þess hversu auðvelt er að búa til, deila, geyma og vinna með stafræna tónlist. Fyrir vikið þarf að þróa aðferðir við að geyma og skoða slík söfn sem nota lýsigögn sem byggja á innihaldi laganna, til viðbótar við lýsigögn eins og heiti laganna og flytjendur. Tegund tónlistarinnar er dæmi um slík lýsigögn. Núna ná bestu aðferðir við sjálfvirka tegundarflokkun tónlistar góðum niðurstöðum, eða um 91% flokkunarnákvæmni. Þessar aðferðir nota hins vegar flókin flokkunarreiknirit (sem eru því hægvirk) sem metin eru með stöðluðum tónlistarsöfnum (sem eru lítil). Það er því erfitt að nota slíkar aðferðir á raunveruleg vandamál, þar sem tónlistarsöfnin eru stærðargráðum stærri.

Þess í stað fjallar þessi skýrsla um tegundarflokkun með k-NN flokkunaraðferðinni. Helsta ástæðan fyrir því að nota svo einfalda flokkunaraðferð er að hún er miklu hraðvirkari en allar aðrar helstu flokkunaraðferðir, sem er nauðsynlegt til að ráða við stór söfn. Hingað til hafa gæði k-NN flokkunar verið talin óásættanleg af vísindamönnum. Við teljum hins vegar að skalanleiki k-NN flokkunar sé næg ástæða til að vinna í að bæta flokkunarhæfni aðferðarinnar. Sem fyrsta skref í þessa átt, sýnum við í þessari skýrslu að k-NN flokkun getur í raun náð býsna góðum árangri við tegundarflokkun tónlistar. Ennfremur sýna afkastamælingar að k-NN flokkun er mjög hraðvirk, sem er nauðsynlegt fyrir raunveruleg gagnasöfn.

Lykilorð: Flokkun tónlistar; Skalanleiki; k-NN flokkun.

Contents

1 Introduction

Today, the majority of all music is created, stored, and played in digital format. A personal music collection may contain thousands of songs, while professional collections typically contain tens or hundreds of thousands. Furthermore, collections grow very fast, partly because file sharing is easy. Browsing and searching large song collections requires new tools as querying by artist, album name or song title becomes inadequate at that scale. Information based on the *contents* of songs, however, is likely to be useful, allowing genre, mood, rythm, key, or even humming to be part of modern search and browsing tools.

In particular, classifying songs according to their genre has proved to be useful, and music genres, such as "pop", "rock", and "jazz" are typical browsing aids for music. As a consequence, automatic genre classification has received much attention lately (e.g., see [BCE+06, LOL03, PBK08, PKA09, TC02]). This work, however, has largely focused on small collections—the collections mainly used contain 1,000–1,500 songs—and the genre labels assigned to songs in these collections are generally quite trustworthy.

1.1 Genre Classification in the Real World

Using state-of-the-art solutions for genre classification in the real world, however, raises several major issues.

First, there is a trend towards assigning multiple labels per song, allowing for a much finer classification [BCE+06]. As a result, the increased number of low-level features to process challenges the complexity of current classification algorithms that are becoming too costly at such a scale. Indeed, Bergstra et al. [BCE+06] could not use the most fine-grained settings to classify the MIREX 2005 collection in less than 24 hours, due to the cost of their approach.

Second, as mentioned above, typical collections are one or two orders of magnitude larger than the ones used when evaluating state of art solutions. More songs obviously produce more features to deal with, which challenges even further the complexity of current approaches.

Last, the vast majority of songs does not have a consistent genre label. From a nicely annotated small collection that a particular algorithm classifies well, it is expected that genre labels will be propagated in such a way that the resulting classification will be of high quality. This remains to be proven, however, as no studies addressing quality issues at a large scale have been performed so far, mainly because the algorithms are too costly and do not scale.

1.2 k-NN Classification

Classifying songs according to genre can be cast into query processing for large collections of high-dimensional vectors. It has already been demonstrated, albeit within the contect of content-based image and video retrieval, that approximate k-nearest neighbor (k-NN) algorithms can cope with scale. Approaches such as Locality Sensitive Hashing (LSH) by Andoni et al. [AI08], the NV-Tree by Lejsek et al. [LAJA09], and visual word approaches such as the one by Philbin et al. [PCI$^+$07], all return high-quality results when indexing hundreds of millions of high-dimensional descriptors. For moderate collections, a carefully implemented exhaustive sequential scan can even perform quite well.

While k-NN genre classification has been studied in the past, the conclusion has invariably been that the classification quality is lower than what could be achieved by other means. Given the lack of efficient implementations of k-NN search at the time these studies were done, this approach to classification has generally been considered useless. This, however, is not true anymore since all these efficient approximate k-NN algorithms have been proposed.

Furthermore, in addition to handling large collections of fine-grained low-level features well, k-NN classification has the added advantage of handling fuzzy genre assignments very well, as different near neighbors can give weights to different genres. For all these reasons, we believe that k-NN classification should be revisited, in order to enable the classification of real-life music collections.

1.3 Contributions

In this report, we describe the initial steps towards automatic classification of large-scale collections of music. The goal of this report is to show, using a standard reference collection and relatively simple software, that k-NN classification should indeed be revisited.

In this report, we therefore study the efficiency and effectiveness of k-NN classification using the collection of Tzanetakis and Cook [TC02]. This collection has been studied well in the past, and the most recent methods achieve over 90% classification accuracy. Since we have not obtained access to the vector generation software of those works, however, we work with the open-source MARSYAS software, and use the resulting vectors for the classification.

Since the scale of the collection is very moderate, we use an optimized sequential scan for query processing. We obtain over 80% classification accuracy, which ranks quite high in the literature. Most importantly, however, those results were obtained in only 3.5 hours on regular hardware, or using about 12 seconds per song. These results indicate very strongly that k-NN classification is indeed a very promising approach for large-scale classification.

The next step in our project is then to apply approximate k-NN classification to a real-life collection of over 50,000 songs, with very fuzzy genre labels. Working with such a collection is far beyond the capabilities of any state-of-the-art algorithms, other than k-NN classification.

Reference	Year	Accuracy
Panagakis et al. [PKA09]	2009	91.0%
Bergstra et al. [BCE$^+$06]	2006	82.5%
Li et al. [LOL03]	2003	78.5%
Panagakis et al. [PBK08]	2008	78.2%
Lidy and Rauber [LR05]	2005	76.8%
Benetos and Kotropoulos [BK08]	2008	75.0%
Holzapfel and Stylianou [HS08]	2008	74.0%
Tzanetakis and Cook [TC02]	2002	61.0%

Table 1: Classification results in the literature.

The remainder of the report is organized as follows. In Section 2 we briefly review related work on classification, and then provide a description of the MARSYAS feature extraction in Section 3. In Section 4 we describe the results of our experiments, before concluding in Section 5.

2 Related Work

Automatic Genre Classification of music has received considerable attention in the music information retrieval community for the last decade. Here we will mention only a few key works; the evolution of classification accuracy is summarized in Table 1 (reproduced from [PKA09]).

The ground-breaking work of Tzanetakis and Cook [TC02] is particularly well known. They classified songs into 10 musical genres based on timbral features, rhythmic content and pitch content. One 30-dimensional feature vector was extracted for each of the 30 seconds long song excerpts in their dataset, which consists of 1,000 excerpts. An accuracy of 61% was achieved using a Gaussian Mixture Model classifier and 60% using k-NN. The dataset used in their paper has since been used by several researchers, and is indeed the set we use for the experiments in this report.

Li et al. [LOL03] used the same features as Tzanetakis and Cook, with the addition of Daubechies Wavelet Coefficient Histograms (DWHC). Wavelets decompose the audio signal based on octaves with each octave having different characteristics. Support Vector Machine classification yielded 78.5% classification accuracy on Tzanetakis dataset while only 62.1% was achieved using k-NN. One feature vector per song was used for the experiments.

Bergstra et al. [BCE$^+$06] won the first prize in the genre classification part of the MIREX 2005 international contest in music information extraction. The genre classification task used two datasets, with approximately 1,500 full length songs in each dataset. Instead of using just one vector for each song, Bergstra et al. extracted features in frames of 1,024 samples and aggregated m such frames into segments before using AdaBoost for classification. This approach yielded 82.3% classification accuracy on MIREX 2005 using $m = 300$, which corresponds to each segment being 13.9 seconds. They achieved 82.5% accuracy on the Tzanetakis collection using the same settings. They state that the best results are achieved by using values of m between 50 and 100, but that this would not have allowed them to complete the task in 24 hours, which was a requirement of the contest.

The ISMIR2004 dataset is another reference collection, which consists of 1,458 full songs using 6 genre classes. Panagakis et al. [PBK08] achieved 78.2% accuracy with the Tzanetakis dataset and 80.9% on the ISMIR2004 dataset using "multiscale spectro-temporal modulation features", where each audio file was converted to a third-order feature tensor generated by a model of auditory cortical processing. Non-negative matrix factorization was used for dimensionality reduction and SVM for classification.

The highest accuracy results for both the Tzanetakis and ISMIR2004 datasets for automatic genre classification were achieved by Panagakis et al. [PKA09] who this time managed to achieve 91% accuracy on the Tzanetakis dataset and 93.6% accuracy on the ISMIR2004 dataset. As before, the authors focused on how audio information is encoded in the human auditory system, and extracted auditory temporal modulation representation of each song. Several classifiers were examined, including support vector machines and k-NN. However, their best results are achieved using sparse representation-based classification (SRC). Their results using k-NN were below 70% for the Tzanetakis dataset.

What these methods generally have in common, is that they use one vector, or a few, to represent each song. It is well known from the image retrieval community, that this is insufficient for image recognition (e.g., see [Low04]), and we believe that further improvements can possibly be made by using multiple descriptors. This, of course, requires more scalable classification methods. Furthermore, classification time is very rarely reported. The only reference that mentions the time necessary for classification is Bergstra et al. [BCE$^+$06], who mention that they could not use the settings they would like if they were to finish the task in under 24 hours. For comparison, we obtain reasonably similar results with k-NN classification in approximately 3.5 hours.

3 MARSYAS

MARSYAS (Music Analysis, Retrieval and Synthesis for Audio Signals) is an open-source software framework for audio analysis and synthesis, with specific emphasis on building Music Information Retrieval systems. The system was first introduced by Tzanetakis and Cook in 1999 [TC99] and used in their influential 2002 paper [TC02]. The framework (available at http://marsyas.sness.net/) includes a variety of modules for common audio tasks. Some representative examples are: feature extraction, soundfile IO, audio IO, signal processing and machine learning.

3.1 Machine Learning

MARSYAS provides a Machine Learning module called "kea", which offers the following three classifiers:

Support Vector Machines (SVM). This popular classifier searches for a hyperplane that separates the data with the maximum margin.

Simple Gaussian (GS). This classifier is based on a multidimensional probability density function defined by mean and variance vectors.

ZeroR. This assigns everything to the most common class, and is therefore not considered.

3.2 Feature Extraction

The feature extraction module "bextract" produces, by default, the following timbral based features:

Time Domain Zero Crossings. Measures the noisiness of the sound by computing the number of sign changes in the time domain.

Spectral Centroid. Measures the shape or brightness of the audiofile by calculating the weighted average frequency of every time frame.

Spectral Flux. The squared change in normalized amplitude between two consecutive time frames and therefore measures how much the sound changes between frames.

Spectral Rolloff. The frequency below which 85% of the energy distribution is achieved.

Mel-Frequency Cepstral Coefficients (MFCC). These are perceptually motivated features which have traditionally been used in speech recognition. MARSYAS stores 13 coefficients.

The following additional features can also be extracted:

Spectral Flatness Measure (SFM). Measures the noisiness of an audio signal. High values indicate high noise levels where all frequencies have similar amount of power, while low values indicate "cleaner" sound.

Spectral Crest Factor (SCF). Another measure of noisiness, closely related to SFM.

Line Spectral Pair (LSP). Represents a compressed version of the signal by two polynomials with alternating roots.

Linear Prediction Cepstral Coefficients (LPCC). This feature represents the evolution of the signal by using a linear combination of recent samples.

CHROMA. A pitch based feature which projects the frequency spectrum into 12 bins, with one bin for each of the 12 distinct pitches of the chromatic musical scale.

In Section 4, we show how the above features affected the quality of our classification.

3.3 Feature Extraction

In order to extract features, the audio signal is broken into short *analysis windows* or time frames, which can possibly overlap. Both the size of the analysis window, w, and the hop size, h, can be changed from the default of 512 samples. The actual features are then calculated as running means and variances over several such analysis windows. This forms another window, called *texture window*, which can be thought of as a memory of the last m descriptors (by default, $m = 40$).

It is worth noting that the "bextract" module looks at the collection of audio files to be analyzed as one big audio file. This has the effect that texture windows can cross the boundaries of audio files, resulting in the first $m - 1$ descriptors of audio file n containing information from audio file $n - 1$. This may affect the recall scores, either positively or negatively, and these descriptors should be discarded or ignored.

4 Genre Classification Experiments

In this section, we report on our classification experiments using a simple k-NN classifier. We start by describing the experimental setup. Then we analyze various aspects of the descriptor configuration, before taking a last look at the classification results.

4.1 Experimental Setup

For our experiments we used the Tzanetakis dataset which consists of 1,000 song excerpts, where each excerpt is 30 seconds long. Each song is sampled at 22,050 KHz in mono. The songs are evenly distributed into the following 10 genres: Blues, Classical, Country, Disco, HipHop, Jazz, Metal, Pop, Reggae and Rock.

We built a k-NN search program which performs a sequential scan of all database descriptors and calculates the Manhattan distance between each query vector and each database vector. Manhattan distance can be implemented using very fast intrinsic SSE code and was therefore selected rather than Euclidean distance.

We used the bextract module of the MARSYAS framework for feature extraction. We then used randomized and stratified 10-fold cross-validation. Stratified randomization is implemented by shuffling songs within each genre, and then concatenating the genre files. 10-fold cross validation is implemented by ignoring, when computing distances, all descriptors from songs that are located within the same 10% of the collection as the query. Furthermore, we ignore the first m feature vectors of each song in the voting process, to account for the overlap between songs.

Table 2 shows the parameters studied and their default values. All the experiments reported were performed on an Apple MacBook Pro machine with 2.66GHz Duo Core Intel processors, 4GB of main memory and one 300GB 5.4Krpm Serial ATA hard disk.

Parameter	**Abbreviation**	**Default Value**
Window Size	w	512
Hop Size	h	512
Memory Size	m	40
Neighbors Retrieved	k	1

Table 2: Parameters of the performance study.

4.2 Comparison to the MARSYAS SVM Classifier

In this section we compare our k-NN classifier to the SVM classifier of MARSYAS. Support vector machines are quite popular among MIR researches, so it is important that our solution is comparable. There are more efficient SVM classifiers available elsewhere, but the result give an indication of the relative efficiency of our solution.

Note that this experiment is different from the rest in three ways. First, it is not randomized and the first m descriptors are not ignored, leading to artificially high accuracy scores for the parameters used. Second, in order to run the SVM classifier is a reasonable time, we set $w = h = 2,048$, in order to reduce the number of descriptors. Third, the accuracy of SVM is reported in a per-descriptor basis, making the results difficult to compare. The main point of this section, however, is the efficiency.

Table 3 shows the accuracy obtained with each of the classifiers, and the total time required to prepare the classifier and classify the 1,000 song excerpts. As the table shows, our simple k-NN classifier achieves better accuracy than the SVM classifier of MARSYAS, using only a fraction of the time.

Note again that in this experiment, the parameters were modified to give a relatively small set of descriptors, such that the SVM package would complete in a reasonable time. Due to the long time required for experiments, the SVM package is not considered further.

4.3 The effect of Window Size, Hop Size and Memory

Starting with the effect of the window size, w, Table 4 shows the impact on accuracy and classification time. We see that accuracy increases as window size gets smaller, until we go from 512 to 256, indicating that 512 (the default value) is indeed the ideal window size.

Classifier	**Accuracy**	**Time**
SVM	67.5%	1,078 min
k-NN	78.1%	10 min

Table 3: Comparison of k-NN and MARSYAS SVM classification (timbral feat., $w = h = 2,048$).

Window Size w	**Accuracy**	**Descriptors**	**Time**
256	77.5%	2,586,641	850.5 min
512	80.4%	1,293,335	207.4 min
1,024	78.6%	647,000	60.5 min
2,048	76.3%	323,834	14.1 min
4,096	72.1%	162,082	3.3 min

Table 4: Effect of window size w (timbral feat. + SFM).

Table 4 furthermore shows the effect on the number of descriptors, and on the running time of the k-NN classification. As expected, the number of descriptors doubles when the window size is halved. Running time, on the other hand, is roughly quadrupled, as both the size of the descriptor collection and the number of query descriptors are doubled.

We also experimented a little with the hop size, h, which impacts the overlap between descriptors, but this did not improve our results. For example, using $w = 512$ and a hop size of $h = 256$ yielded 80.2% accuracy.

Having reached our best accuracy with $w = h = 512$ samples, using timbral features along with SFM, we next experimented with the memory size. Table 5 shows that we improve our accuracy neither by increasing nor decreasing m. Again, the default value is optimal.

4.4 The Effect of Feature Selection

For this experiment we started by extracting the default timbral features. We then experimented with adding features; the results are summarized in Table 6. The table shows that adding Spectral Flatness Measure (SFM) increases the accuracy for this test set from 75.4% to 80.4%. Adding further information to the feature vector, however, did not improve the results, and in fact we observed that it actually hurts the results slightly. We conclude that in addition to the default timbral features it is beneficial to include Spectral Flatness Measure to achieve best results.

Memory Size m	**Accuracy**
30	80.0%
40	80.4%
50	79.2%

Table 5: Effect of memory size m (timbral feat. + SFM).

Features	**Dimensions**	**Accuracy**
Timbral features (TF)	34	75.4%
TF + SFM	82	80.4%
TF + SFM + SCF	130	80.0%
TF + SFM + LSP	118	80.0%
TF + SFM + LPCC	106	79.8%
TF + SFM + CHROMA	106	79.4%
TF + SFM + SCF + LSP	166	79.8%
TF + SFM + SCF + LPCC	154	79.4%
TF + SFM + SCF + CHROMA	154	79.9%

Table 6: Effect of feature selection on accuracy.

4.5 The Effect of Varying k

Finally, we experiment with the value of the k parameter, which indicates how many neighbors from the collection are considered for each query descriptor. We ran the k-NN search for values of k ranging from 1 to 10. Table 7 shows the results. As the table shows, varying k does not affect the classification accuracy much, nor does it have any significant impact on the classification time. Remember that our query set contains all 1,000 songs in the dataset, so changing the accuracy from 80.4% to 80.8% only means that four more songs are found.

4.6 Result Analysis

Table 8 shows the "confusion" matrix for our experiment of table 7, for $k = 3$, where we managed 80.8% accuracy. We see that in most cases the mistakes the program does are similar to those humans commonly make, and in general these results agree with others reported in the literature. The table shows that 3 blues songs are classified as jazz, 7 country songs are classified as pop (which is a very fuzzy classification genre), 10 disco songs are classified as pop, 8 jazz songs are classified as classic, and 7 reggae songs are classified as pop. What is somewhat surprising is the low accuracy of rock song classification, especially the fact that 20 rock songs are classified as disco songs. We intend to investigate the misclassification of rock songs specifically in our future research.

Neighbors k	**Accuracy**	**Time**
1	80.4%	207.4 min
2	80.7%	208.3 min
3	80.8%	209.3 min
4	80.6%	210.1 min
5	80.5%	212.3 min
10	80.5%	217.5 min

Table 7: Effect of varying k (timbral features + SFM).

	Blues	**Classical**	**Country**	**Disco**	**Hiphop**	**Jazz**	**Metal**	**Pop**	**Reggae**	**Rock**
Blues	**93**	0	0	3	0	3	0	0	0	1
Classical	0	**100**	0	0	0	0	0	0	0	0
Country	4	7	**66**	11	0	1	0	7	2	2
Disco	1	1	2	**80**	3	0	0	10	1	2
Hiphop	2	0	1	0	**80**	0	2	8	5	2
Jazz	0	8	0	0	0	**91**	0	1	0	0
Metal	2	0	0	1	1	0	**92**	0	0	4
Pop	1	3	3	2	1	0	1	**87**	1	1
Regggae	1	1	1	8	5	0	0	7	**75**	2
Rock	3	7	3	20	4	3	5	7	4	**44**
Total	107	127	76	125	94	98	100	126	88	58

Table 8: Confusion matrix (timbral features + SFM, $k = 3$).

We performed a small study where 10 musicians were asked to classify 4 songs which were incorrectly classified by our system. The results of this study are shown in Table 9, together with a detailed analysis of the votes of the k-NN classifier. While the detailed information about individual songs must be omitted, due to lack of space, this study provides interesting results despite its lack of coverage. In only one case do the musicians agree with the ground truth. In two cases they agree with the k-NN classifier, and in one case they agree with neither. This suggests that the ground truth is indeed not the absolute truth, and that perhaps the ground truth for the two songs where both k-NN and musicians agree, should be changed. It also shows that, in general, it would be beneficial to be able to use more than one genre for each song, so that when a song receives a similar number of votes from multiple genres we could indeed use them all.

4.7 Summary

To summarize the discussion here, the key point is that using k-NN classification, we can obtain a very respectable result of 80.8% accuracy, which places us third in the quality list of Table 1. While previous results have denounced k-NN classification, this is presumably because those studies focused on using a single descriptor, or very few descriptors, for each song. The fact that we can obtain these results in only 3.5 hours, on very reasonable hardware, leads us to believe that k-NN classification is a viable option at a large scale.

5 Conclusions and Future Work

We have shown, through a number of experiments, that k-NN classification can be used with good results for automatic genre classification. We have also found that adding the Spectral Flatness Measure features to the standard timbral features improves classification accuracy, and that a window

File Name	**Ground Truth**	**k-NN**	**Musicians**
country.00030	Country	Jazz (0.43),Country (0.29), Blues(0.2), …	Country (0.6), Jazz (0.4)
metal.00058	Metal	Rock (1.0)	Rock (1.0)
rock.00249	Rock	Classical (0.33), Pop (0.2), Disco (0.14), …	Pop (0.7), Rock (0.2), Country (0.1)
rock.00058	Rock	Pop (0.27), Disco (0.18), Country (0.18), …	Pop (0.8), Classical (0.2)

Table 9: Results from the ground truth experiment. The number in parentheses indicates fraction of votes for each genre.

size of 512 yields the best classification results. More importantly, however, k-NN based classification is very fast compared to other state-of-the-art genre classification schemes. This property is crucial as genre classification will eventually be applied to real life collections containing hundreds of thousands of songs.

Efficient and scalable algorithms for high-dimensional k-NN searches have been developed in the computer vision and databases communities. The quality of k-NN based classification, however, is currently about 10% below the best results in the literature. Since the efficiency of k-NN schemes make them good candidates for low-level support for classification, it is necessary to understand the fundamental reasons for this difference. We need to understand whether the difference in quality arises from fundamental problems with k-NN classification, or whether different features are required for k-NN classification than for other approaches. For example, we have certainly shown in this report that k-NN classification works best with a large number of features per song. Investigating these issues is part of our future work, such as experimenting with the various combinations of acoustic features that have proven to be work well for classifying genres; our goal is to reach quality levels quite similar to state-of-the-art solutions.

While it is known that k-NN schemes work at scale, another important line of work is to determine whether the current approaches for scaling other classifiers, such as SVM (e.g., see [DNP08]), can work well in the context of large-scale genre classification. High-dimensional approaches that cope with scale (regardless of whether they are based on SVM or k-NN) are almost all approximate, with precision being traded-off against dramatic performance improvements. Therefore, it is a very interesting direction for future work to determine whether the resulting lower precision is acceptable for real-life collections, and whether tuning parameters, inventing new features, or inventing new ways to combine features may help to compensate for that loss. Certainly, continuing to optimize the accuracy for small scale reference collections is a dead end.

6 Acknowledgement

We would like to thank George Tzanetakis for providing his genre classification dataset, and for his helpful assistance with the MARSYAS software.

References

[AI08] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Com. of the ACM*, 51(1), 2008.

[BCE+06] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and adaboost for music classification. *Machine Learning*, 65(2-3), 2006.

[BK08] E. Benetos and C. Kotropoulos. A tensor-based approach for automatic music genre classification. In *Proc. EUSIPCO*, Lausanne, Switzerland, 2008.

[DNP08] T.-N. Do, V.-H. Nguyen, and F. Poulet. Speed up SVM algorithm for massive classification tasks. In *Proc. Advanced Data Mining and Applications*, Chengdu, China, 2008.

[HS08] A. Holzapfel and Y. Stylianou. Musical genre classification using nonnegative matrix factorization-based features. *IEEE Trans. on Audio, Speech & Language Processing*, 16(2), 2008.

[LAJA09] H. Lejsek, F. H. Ásmundsson, B. Þ. Jónsson, and L. Amsaleg. Nv-tree: An efficient disk-based index for approximate search in very large high-dimensional collections. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(5), 2009.

[LOL03] T. Li, M. Ogihara, and Q. Li. A comparative study on content-based music genre classification. In *Proc. SIGIR*, 2003.

[Low04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal on Computer Vision*, 60(2), 2004.

[LR05] T. Lidy and A. Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proc. ISMIR*, London, UK, 2005.

[PBK08] Y. Panagakis, E. Benetos, and C. Kotropoulos. Music genre classification: A multilinear approach. In *Proc. ISMIR*, Philadelphia, USA, 2008.

[PCI+07] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, Mineapolis, MN, USA, 2007.

[PKA09] Y. Panagakis, C. Kotropoulos, and G. R. Arce. Music genre classification via sparse representations of auditory temporal modulations. In *Proc. EUSIPCO*, Glascow, Scotland, 2009.

[TC99] G. Tzanetakis and P. Cook. Marsyas: a framework for audio analysis. *Organised Sound*, 4(3), 1999.

[TC02] G. Tzanetakis and P. R. Cook. Musical genre classification of audio signals. *IEEE Trans. on Speech & Audio Processing*, 10(5), 2002.

Technical Report RUTR-CS10001
ISSN 1670-5777

School of Computer Science
Reykjavík University
Menntavegi 1
101 Reykjavík, Iceland
Tel. +354 599 6200
Fax +354 599 6201
www.ru.is