Using Fisher Kernels and Hidden Markov Models for the Identification of Famous Composers from their Sheet Music

David R. Hardoon
University of Southampton
ISIS Research Group
Building 1, Highfield
Southampton, SO17 1BJ
drh@ecs.soton.ac.uk

Craig Saunders
University of Southampton
ISIS Research Group
Building 1, Highfield
Southampton, SO17 1BJ
cjs@ecs.soton.ac.uk

John Shawe- Taylor
University of Southampton
ISIS Research Group
Building 1, Highfield
Southampton, SO17 1BJ
jst@ecs.soton.ac.uk

ABSTRACT

We present a novel application of Fisher kernels to the problem of identifying famous composers from their sheet music. The characteristics of the composers writing style are obtained from note changes on a basic beat level, combined with the notes hidden harmony. We are able to extract this information by the application of a Hidden Markov Model to learn the underlying probabilistic structure of the score. We are able to use the model to generate new sheet music based on a specific composer. Furthermore we do an identification comparison using Fisher kernels and a Hidden Markov Model on limited data.

Keywords: Fisher Kernel, HMM, SVM, Composer Identification.

1 Introduction

In the last several years music information retrieval, generation and classification has become a major topic of interest. The analysis of music has been addressed using natural language modelling and statistical models that have been mostly focused on digital audio resources. Batlle and Cano (2000) have applied Hidden Markov Models (HMM)(Rabiner, 1989) on audio fragments for music segmentation and classification (Batlle et al., 2004). While others have applied grammatical induction to learn the genre and style of MIDI files. An interesting review of music generation from statistical models is given by Conklin (2003).

In previous work (Saunders et al., 2004) we have presented a novel application of string kernels to the problem of recognising famous pianists from their style of playing. This was done using a measurement of the changes in beat-level tempo and beat-level loudness (Zanon and Widmer, 2003) from audio recordings of the performers playing. In (Saunders et al., 2004) we assume that a performer may consistently produce loudness/tempo changes unique to themselves at specific points in a piece. In this paper we further continue this prior assumption to the problem of identifying famous composers using minimal information obtained from their sheet music. We assume that a composer may consistently write changes in note combination, tempo, and underlying harmony unique to their style. The problem of identifying a personal style in the written score is far more complex then the identification of style in an actual performance, as different genres of music composition have specific rules of which a composer usually adheres to.

We propose a HMM for sheet music. The key concept of HMM theory is, given a sequence of observed events to find the most probably hidden state sequence of the observed sequence. In our model the hidden state is the underlying harmony. An application of example-based music generation using Markov Chains on notes was done by Kohonen et al. (1991). Furthermore we use the Fisher kernel for the identification task, as the underlying concept is that the fisher score, which measures the change a probabilistic model must go through in order to accommodate a new sequence, will be able to extract from the model more information than given by their output probabilities alone. In our experimentation we compare the application of the two approaches on limited data.

The papers outline is as follows; In the following section we provide a description of our representation of sheet music and a description of the HMM used. Section 3 outlines the Fisher score and Fisher kernels algorithm. We then present experimental set up and results in section 4 and conclude in section 5 raising suggestions for future research.

2 Hidden Markov Model for Sheet Music

Using Mup 5.0 software by Arkkra Enterprises[1] to provide us with a machine readable format of sheet music. Mup takes a text file as input and produces PostScript output for printed music. It can handle both regular notation and tablature notation. Mup can also produce MIDI output and by using a third party extension to Mup, *midi2mup*[2], one can convert MIDI files to Mup files.

We represent the musical score by breaking down

[1] http://www.arkkra.com/

[2] http://www.arkkra.com/doc/userpgms.html

the length of a bar into 16 states (i.e. notes), each state consists of the length of a $\frac{1}{16}$'th of a beat. The choice of $\frac{1}{16}$ is arbitrary as we believe it to be of sufficient low-level information to capture the common changes that occur. In order to handle notes of a longer tempo, we keep track of the number of times a note needs to be repeated in order to "complete" its actual length.

The model consists of four nodes in total; L - the current Location in the bar, H - the underlying Harmony of the played note, N - the played Note and D - the Duration of current played note. As the scores underlying harmony is unknown, H is defined to be a hidden node consisting of 24 states, 12 for the major and 12 for the minor possibilities. N is an observed node with 13 states, 12 states for each note and the 13'th state representing a rest. We only have 12 possible played notes in the model as we reduce all octaves to a single one. L is an observed node with 16 states, one for each position in the bar. D is an observed node of 16 states representing the duration of the current note played, a value of 1 represents a note being played for $\frac{1}{16}$ of a beat and a value of 16 represents a note being played for the duration of the whole bar (i.e. $16 * \frac{1}{16} = 1$). As we have defined our model to have 16 notes per bar; a note $n \in N$ played for the length, for say, $\frac{1}{4}$ in position one will be in fact the same note repeated four times with a reducing duration, as shown as follows;

$$\begin{aligned} D_1 = 4,\ L_1 = 1, N_1 = n \\ D_2 = 3,\ L_2 = 2, N_2 = n \\ D_3 = 2,\ L_3 = 3, N_3 = n \\ D_4 = 1,\ L_4 = 4, N_4 = n. \end{aligned}$$

We define a model that would be both generic such that it would be applicable for different types of scores and yet still specific enough to be able to extract the changes along the underlying harmony. We define the following model using the nodes described above. The location of

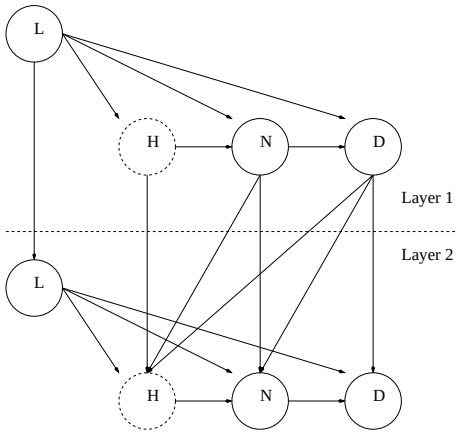

Figure 1: HMM for Sheet Music.

a note is only dependent on the previous location, as we can only move from location 1 to location 2 and so forth until location 16 which then the location sequence is repeated. We believe that the underlying hidden harmony is depended on a combination of the current location in the bar, the previous harmony, the previous note played and its actual playing duration. The current note is dependent on the current harmony in which we are in, the location in the bar and the previous note played with its duration. While the current duration of a played note is dependent on the current played note, its location in the bar and the previous duration of the previous note. In figure 1 a graphical example of the dependencies between two layers of the model is presented, these are also defined formally as follows

$$\begin{aligned} P(L_i|L_{i-1}) \\ P(H_i|L_i, D_{i-1}, N_{i-1}, H_{i-1}) \\ P(N_i|H_i, L_i, D_{i-1}, N_{i-1}) \\ P(D_i|N_i, L_i, D_{i-1}) \end{aligned}$$

were we use, for simplicity, the convention that for $i = 1$

$$\begin{aligned} P(L_1|L_0) &= P(L_1) \\ P(H_1|L_1, D_0, N_0, H_0) &= P(H_1|L_1) \\ P(N_1|H_1, L_1, D_0, N_0) &= P(N_1|H_1, L_1) \\ P(D_1|N_1, L_1, D_0) &= P(D_1|N_1, L_1). \end{aligned}$$

We are able to claim with absolute certainty that when $D_i > 1$ that $D_{i+1} = D_i - 1$, $N_{i+1} = N_i$ and $H_{i+1} = H_i$, as a note will not change while it is actually being played. Therefore we set the models probabilities to uphold this condition a priori to the training of the model.

3 Fisher Scores and Kernels

The idea of generating a feature representation and associated kernel from a probabilistic model of our data is very appealing. It suggests a practical way of incorporating domain knowledge while still allowing us to use powerful non-parametric methods such as support vector machines. In this sense we can hope to get the best of both worlds: use the probabilistic models to create a feature representation that captures our prior knowledge of the domain, while leaving open the actual analysis methods to be used.

The most popular method of deriving a feature vector from a probabilistic model is known as the Fisher score or Fisher kernel. This method creates a feature vector from a smoothly parametrised probabilistic model by computing the gradients of the log-likelihood of a data item around the chosen parameter setting. This can be seen as estimating the direction the model would be deformed if we were to incorporate the data item into the parameter estimation.

We now give formal definitions of these concepts.

Definition 1 [Fisher score and Fisher information matrix] The *log-likelihood* of a data item x with respect to the model $m(\theta^0)$ for a given setting of the parameters θ^0 is defined to be

$$\log \mathcal{L}_{\theta^0}(x),$$

where for a given setting of the parameters θ the likelihood is given by

$$\mathcal{L}_\theta(x) = P(x|\theta) = P_\theta(x).$$

Consider the vector gradient of the log-likelihood

$$\mathbf{g}(\theta, x) = \left(\frac{\partial \log \mathcal{L}_\theta(x)}{\partial \theta_i} \right)_{i=1}^{N}.$$

The *Fisher score* of a data item x with respect to the model $m(\theta^0)$ for a given setting of the parameters θ^0 is

$$\mathbf{g}\left(\theta^0, x\right).$$

The *Fisher information matrix* with respect to the model $m(\theta^0)$ for a given setting of the parameters θ^0 is given by

$$\mathbf{I}_M = \mathbb{E}\left[\mathbf{g}\left(\theta^0, x\right)\mathbf{g}\left(\theta^0, x\right)'\right],$$

where the expectation is over the generation of the data point x according to the data generating distribution. ■

The Fisher score gives us an embedding into the feature space $\mathbb{R}^N$ and hence immediately suggests a possible kernel. The matrix $\mathbf{I}_M$ can be used to define a non-standard inner product in that feature space.

Definition 2 [Fisher kernel] The invariant Fisher kernel with respect to the model $m(\theta^0)$ for a given setting of the parameters θ^0 is defined as

$$\kappa(x, z) = \mathbf{g}\left(\theta^0, x\right)' \mathbf{I}_M^{-1} \mathbf{g}\left(\theta^0, z\right).$$

The practical Fisher kernel is defined as

$$\kappa(x, z) = \mathbf{g}\left(\theta^0, x\right)' \mathbf{g}\left(\theta^0, z\right).$$

■

We will follow the standard route of restricting ourselves to the practical Fisher kernel. Notice that this kernel may give a small norm to very typical points, while atypical points may have large derivatives and so a correspondingly large norm. There is, therefore, a danger that the inner product between two typical points can become very small, despite their being similar in the model. This effect can be overcome by normalising the kernel. Another way to use the Fisher score that does not suffer from the problem described above is to use the Gaussian kernel based on distance in the Fisher score space

$$\kappa(x, z) = \exp\left(-\frac{\|\mathbf{g}\left(\theta^0, x\right) - \mathbf{g}\left(\theta^0, z\right)\|^2}{2\sigma^2}\right).$$

For a detailed review of Fisher score and kernel see Shawe-Taylor and Cristianini (2004).

We will be using fixed length hidden Markov models. For these the Fisher scores can be computed as shown in Figure 2. Note that we have only computed the Fisher scores for the emission probabilities $P(\sigma|a)$ for state a and symbol σ. Further work will consider incorporating the Fisher scores associated with the transition probabilities $P_M(b|a)$. Note that in our application the state variable incorporates the information of the hidden state as well as the bar position. This simple trick allows us to handle the apparently more general situation of allowing different emission probabilities for different positions in the sequence.

Input	Symbol string s, state transition probability matrix $P_M(a\|b)$, initial state probabilities $P_M(a) = P_M(a\|a_0)$ and conditional probabilities $P(\sigma\|a)$ of symbols given states.
Process	Assume p states, $0, 1, \ldots, p$.
2	$\mathrm{score}(:,:) = 0;$
3	$\mathrm{forw}(:,0) = 0;$
4	$\mathrm{back}(:,n) = 1;$
	$\mathrm{forw}(0,0) = 1; \mathrm{Prob} = 0;$
5	for $i = 1 : n$
7	for $a = 1 : p$
8	$\mathrm{forw}(a,i) = 0;$
9	for $b = 0 : p$
10	$\mathrm{forw}(a,i) = \mathrm{forw}(a,i) + P_M(a\|b)\,\mathrm{forw}(b,i-1);$
11	end
12	$\mathrm{forw}(a,i) = \mathrm{forw}(a,i)\,P(s_i\|a);$
13	end
14	end
15	for $a = 1 : p$
16	$\mathrm{Prob} = \mathrm{Prob} + \mathrm{forw}(a,n);$
17	end
18	for $i = n-1 : 1$
19	for $a = 1 : p$
20	$\mathrm{back}(a,i) = 0;$
21	for $b = 1 : p$
22	$\mathrm{back}(a,i) = \mathrm{back}(a,i) + P_M(b\|a)\,P(s_{i+1}\|b)\,\mathrm{back}(b,i+1);$
23	end
24	$\mathrm{score}(a,s_i) = \mathrm{score}(a,s_i) + \mathrm{back}(a,i)\,\mathrm{forw}(a,i)\,/\,(P(s_i\|a)\,\mathrm{Prob});$
25	for $\sigma \in \Sigma$
26	$\mathrm{score}(a,\sigma) = \mathrm{score}(a,\sigma) - \mathrm{back}(a,i)\,\mathrm{forw}(a,i)\,/\mathrm{Prob};$
27	end
Output	Fisher score $= \vec{\mathrm{score}}$

Figure 2: Pseudocode to compute the Fisher scores for the fixed length Markov model Fisher kernel.

4 Experiements

4.1 Experimental Setup

In our experiments we use the MIDI database from the school of music in the University of Arizona[3] for MIDI files of composition by *Johann Sebastian Bach, Wolfgang Amadeus Mozart, Ludwig van Beethoven* and *Georg Frideric Handel*. We extract the following MIDI files for each composer:

Johann Sebastian Bach -
Brandenburg Concerto #2 in F (I)
Brandenburg Concerto #2 in F (II)
Brandenburg Concerto #2 in F (III)

Wolfgang Amadeus Mozart -
Sonata K. 545 (I)
Symphony #40 in g (III)

[3] http://www.arts.arizona.edu/midi/

Ludwig van Beethoven -
Symphony #6 in F (I)

Georg Frideric Handel -
Air from Keyboard Suite V in E major

Although this data is limited in size, it is the only publicly available data we were able to obtain that could be converted via Mup. The MIDI files were converted into machine readable sheet music using Mup. Then the score's for each composer were combined and divided sequentially into separate samples, where each sample contained equally 7 bars. Therefore we obtain 18 samples for *Johann Sebastian Bach* where 14 are used as training samples and 4 as testing samples. 14 samples for *Wolfgang Amadeus Mozart* where 10 are used as training samples and 4 as testing samples. 14 samples for *Ludwig van Beethoven* where 10 are training samples and 4 are testing samples and 11 for *Georg Frideric Handel* where 8 are using as training samples and 3 as testing samples. In order to keep the methods comparable we use the same training-testing sample split across the different approaches.

In the following section we compute per method the Receiver Operating Characteristic (ROC) true-positive and false-positive values using Matlab[4] code written by Fawcett (2004). This allows us to compare results across the two different methods. We also compute the area under the ROC using the same package by Fawcett (2004). The HMM model was implemented in Matlab using the Bayes Net Toolbox written by Kevin Murphy[5]. We have also used the OSU-SVM 3.00 toolbox for Matlab[6] implementation of support vector machines (for more information regarding SVM's see Cristianini and Shawe-Taylor (2000)) with the set parameter settings, for the Linear kernel penalty parameter $C = 11$ and for the Gaussian kernel penalty parameter $C = 10$ and Gaussian width $(\frac{1}{2\sigma^2}) = \gamma = 0.1$. These fixed settings may not produce the best possible results, although we are unable to tune our parameter due to our limited data.

4.2 Music Generation

Following the training process of the individual models per composer, we are able to generate sequences of "new" sheet music using the learnt probabilities within the model. Demonstrating that the models had indeed learnt a particular relation of the notes as written in the sheet music.

We generate and present new sheet music as follows; In figure 3 for *Wolfgang Amadeus Mozart*. In figure 4 for *Johann Sebastian Bach*. In figure 5 for *Ludwig van Beethoven* and in figure 6 for *Georg Frideric Handel*.

Although several obvious harmonic errors in the generated score's in figures 3−6 we are able to observe a clear distinction between the individual composers as depicted in the actual generated sequence. It may be interesting in further work to address the "creative" aspect of music generation using HMMConklin (2003).

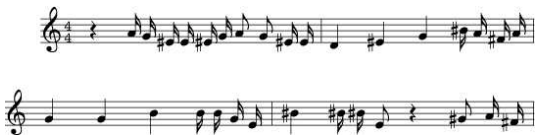

Figure 3: Mozart generated sequence.

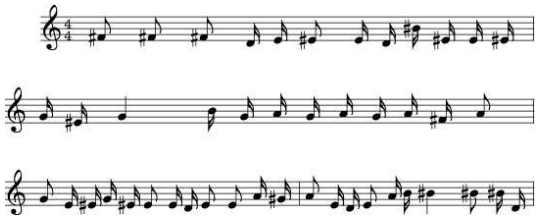

Figure 4: Bach generated sequence.

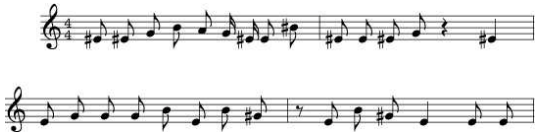

Figure 5: Beethoven generated sequence.

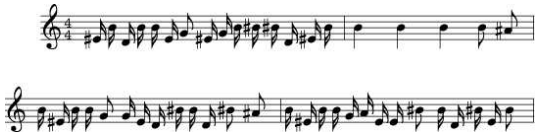

Figure 6: Handel generated sequence.

4.3 Identification Results

For brevity we us the following short hand across the tables in this section;

M represents *Wolfgang Amadeus Mozart*
Ba represents *Johann Sebastian Bach*
Be represents *Ludwig van Beethoven*
H represents *Georg Frideric Handel*.

In the following tables, X-Model refers to the HMM model trained on the training samples of composer X. The results displayed under X-Model are for the testing samples of composer X versus the testing samples of the remaining composers.

We test the classification rate of the Markov models, per composer, by computing the log-likelihood of a testing sample being generated by that particular model. In table 1 the area under the ROC values, as plotted in figures 7−10, for the four HMM models are given. A area

[4] http://www.mathworks.com/
[5] http://www.cs.ubc.ca/∼murphyk/Software/BNT/bnt.html
[6] http://www.ece.osu.edu/∼maj/osu_svm/

under the ROC of 0.5 represents random (50%). We are able to observe that our proposed HMM, excluding the model for *Ludwig van Beethoven*, is indeed able to learn from the sheet music and identify quite efficiently the different composers. It is interesting to observe the lower then random identification result produced for the *Ludwig van Beethoven* model versus *Wolfgang Amadeus Mozart* and *Georg Frideric Handel* while the *Georg Frideric Handel* model and the *Wolfgang Amadeus Mozart* model are able to identify *Georg Frideric Handel* and *Wolfgang Amadeus Mozart* from *Ludwig van Beethoven* with very high result.

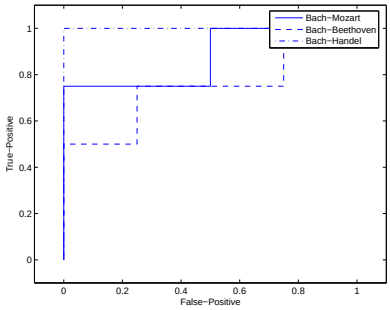

Figure 7: HMM ROC curve for the Bach model.

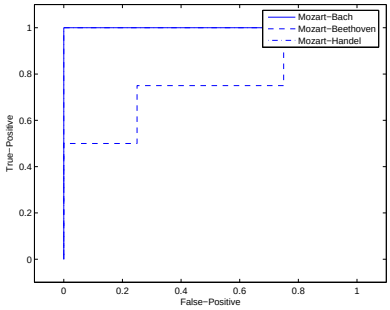

Figure 8: HMM ROC curve for the Mozart model.

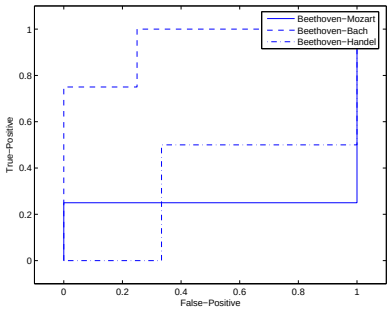

Figure 9: HMM ROC curve for the Beethoven model.

Following section 3 we compute the Fisher scores, per model (i.e. composer), for the training and testing

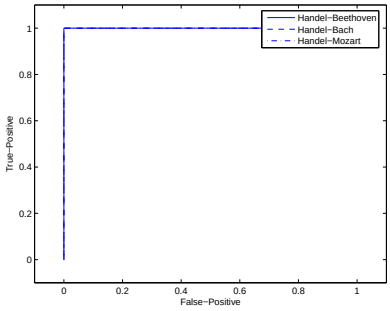

Figure 10: HMM ROC curve for the Handel model.

Table 1: Composer identification area under ROC using the HMM.

vs.	**M-Model**	**Ba-Model**	**Be-Model**	**H-Model**
M		0.88	0.25	1.00
Ba	1.00		0.94	1.00
Be	0.75	0.75		1.00
H	1.00	1.00	0.34	

samples. The training samples are the same samples as used for the HMM. We train our SVM on the training samples fisher scores of the particular composer that the model was trained on, as our positive labelled data. Our negative labelled data for the training process are the training samples fisher scores of the composer that we are comparing against. The testing is the similar process but with the testing samples from both composers.

In table 2 the area under the ROC values, as plotted in figures $11-14$, for the linear kernel on the Fisher score. It seems the the linear kernel of the fisher score is unable to extract the relevant information for the composer classification. Although further research into tuning the SVM penalty parameter C and further experimentation with more data may yield a higher accuracy rate.

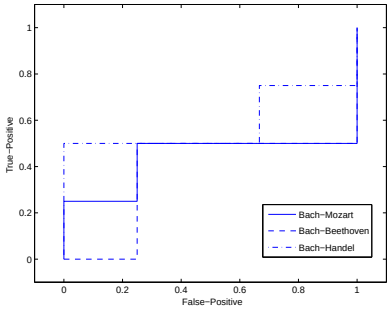

Figure 11: Linear Fisher kernel ROC curve for the Bach model.

In table 3 the area under the ROC values, as plotted in figures $15-18$, for the Gaussian kernel on the Fisher

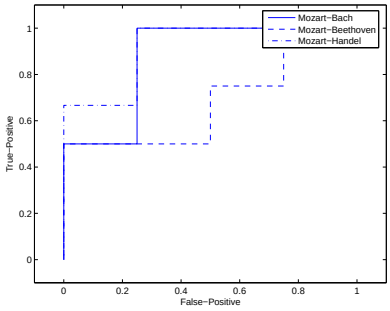

Figure 12: Linear Fisher kernel ROC curve for the Mozart model.

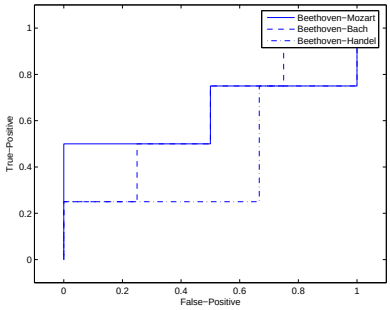

Figure 13: Linear Fisher kernel ROC curve for the Beethoven model.

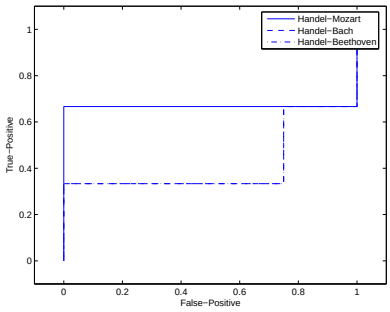

Figure 14: Linear Fisher kernel ROC curve for the Handel model.

Table 2: Composer identification area under ROC using the Fisher Linear kernel.

vs.	M-Model	Ba-Model	Be-Model	H-Model
M		0.44	0.63	0.67
Ba	0.88		0.63	0.42
Be	0.69	0.38		0.42
H	0.92	0.59	0.42	

score. The Gaussian kernel over the fisher score is able to yield a higher classification rate then the linear kernel, although slightly lower then the HMM. Again, as with the Linear kernel, turning the SVM penalty parameter C and Gaussian width parameter γ per model may result in a higher accuracy rate then currently reported.

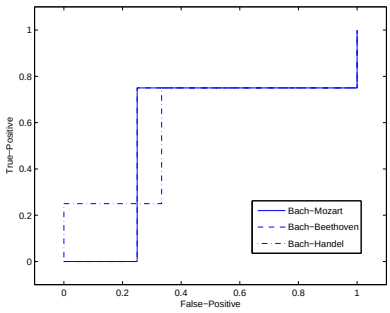

Figure 15: Linear Fisher kernel ROC curve for the Bach model.

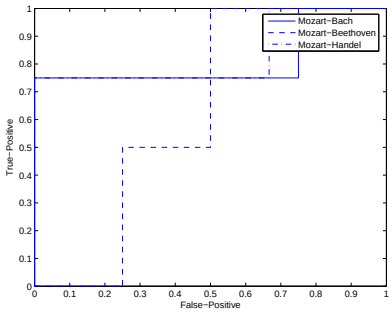

Figure 16: Linear Fisher kernel ROC curve for the Mozart model.

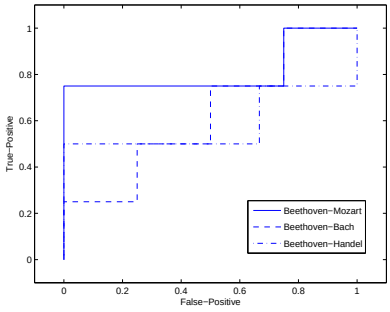

Figure 17: Linear Fisher kernel ROC curve for the Beethoven model.

We are able to observe that by using the HMM and the Gaussian kernel on the fisher score, we are indeed able to identify composers given their sheet music. The results of the Gaussian kernel on the Fisher score suggest that with proper SVM parameter tuning we may be able to outperform the HMM, as the Gaussian kernel seems to be able to extract more information from the *Ludwig van Beethoven* model, achieving a higher area under the ROC value, then

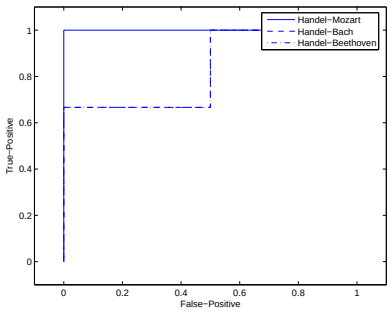

Figure 18: Linear Fisher kernel ROC curve for the Handel model.

Table 3: Composer identification area under ROC using the Fisher Gaussian kernel.

vs.	M-Model	Ba-Model	Be-Model	H-Model
M		0.57	0.82	1.00
Ba	0.82		0.63	0.84
Be	0.63	0.57		0.84
H	0.83	0.59	0.59	

with that with the HMM alone. The results also suggest that the usage of the Linear kernel on the Fisher score is not an efficient one.

5 Conclusions

In this work we have presented the application of applying a HMM probabilistic model for the problem of learning to identify famous composers given their music sheet. We also present a novel application of Fisher kernels for the same problem. Our proposed HMM, although relatively simplistic considering the complexity of the task, has shown to be able to extract characteristics of style sheet music. Our work focuses on the identification on actual written score, while most common work relays on a performance representation of the score for classification. We believe to have presented a novel and interesting solution to a fairly complex and intriguing problem. For future work, we would like to reproduce our work on a much larger scale data, also investigating the SVM parameter tuning for optimising our results.

ACKNOWLEDGEMENTS

This work was supported by the PASCAL Network of Excellence, IST-2002-506778. D.R.H is also supported by the European project LAVA IST-2001-34405. C.S is supported in by the EPSRC grant no GR/S22301/01 (Development and Application of String-Type Kernels).

References

Eloi Batlle and Pedro Cano. Automatic segmentation for music classification using competitive hidden markov models. In *International Symposium on Music Information Retrieval*, Plymouth, MA, USA, 2000.

Eloi Batlle, Jaume Masip, and Enric Guaus. Amadeus: A scalable hmm-based audio information retrieval system. In *First International Symposium on Control, Communications and Signal Processing (ISCCSP 2004)*, Hammamet, Tunisia, 2004.

Darrell Conklin. Music generation from statistical models. In *Symposium on Artificial Intelligence and Creativity in the Arts and Sciences (AISB 2003)*, Aberysywyth, Wales, UK, 2003.

Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

T. Fawcett. ROC graphs: Notes and practical considerations for researchers. Technical Report MS 1143, HP Laboratories, 1051 Page Mill Road, Palo Alto, CA 94304, USA, 2004.

Thomas Hofmann. Learning the similarity of documents: An information-geometric approach to document retrieval and categorization. In *NIPS*, pages 914–920, 1999.

T. Kohonen, P. Laine, K. Tiits, and K. Torkkola. A nonheuristic automatic composing method. *Music and Connectionism*, pages 229–242, 1991.

F. Lerdahl and R. Jackendoff. *A generative theory of tonal music*. M.I.T. Press, 1983.

R. B. Lyngso and C. N. S. Pedersen. Complexity of comparing hidden markov models. In *Proceedings of the ISAAC*, volume 2223 in LNCS, pages 416–428, Bering, Heidelberg, 2001. Springer-Verlag.

Eduardo Miranda, Andrew Brouse, Bram Boskamp, and Hilary Mullaney. Plymouth brain-computer music interface project: Intelligent assistive technology for music-making, (submitted to conference). URL `http://cmr.soc.plymouth.ac.uk`.

Francois Pachet. The continuator: Musical interaction with sytle. *Journal of New Music Research*, 31 (1), 2002.

L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, 1989.

Craig Saunders, David R. Hardoon, John Shawe-Taylor, and Gerhard Widmer. Using string kernels to identify performers from their playing style. In *15'th European Conference on Machine Learning (ECML '04)*, Pisa, Italy, 2004.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

Patrick Zanon and Gerhard Widmer. Learning to recognise famous pianists with machine learning techniques. In *Proceedings of the Stockholm Music Accoustics Conference (SMAC '03)*, August 2003.