Tagging Choreographic Data for Data Mining and Classification

Catalina Anca Ioan, Julien Velcin, Stefan Trausan-Matu

To cite this version:

Catalina Anca Ioan, Julien Velcin, Stefan Trausan-Matu. Tagging Choreographic Data for Data Mining and Classification. 24th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Jul 2013, Athens, Greece. hal-01953434

HAL Id: hal-01953434

https://hal.univ-lyon2.fr/hal-01953434

Submitted on 12 Dec 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Tagging Choreographic Data for Data Mining and Classification

Cătălina - Anca Ioan
"Politehnica" University of Bucharest
Bucharest, Romania
Email: catalina.ioan@cs.pub.ro

Julien Velcin
Laboratoire ERIC,
Université Lumière Lyon 2
Lyon, France
Email: julien.velcin@eric.univ-lyon2.fr

Ştefan Trăuşan - Matu
"Politehnica" University of Bucharest
Bucharest, Romania
Email: stefan.trausan@cs.pub.ro

***Abstract*—We propose an original approach for mapping the choreographic data into a new representation language adapted to data mining techniques. Our approach relies mainly on the notion of "dance tags" that we took from the NLP community by analogy with Part-of-Speech tagging. The process starts from scores described in Labanotation and produces in a fully automatic manner a high-level, comprehensive representation of the choreographic sequence. Our experiments show that we succeed in retrieving manually translated scores with an accuracy of 85% to 94%. Using this new representation of the choreographic data, one can then perform several useful tasks in an efficient manner. Among these are: music recommendation, automated detection of dance style or genre, and ultimately any task that requires a deeper understanding of the meaning of choreographic information than traditional processing can provide. In this paper, we demonstrate the usefulness of our approach with a simple example for discriminating between classical ballet, modern ballet, and folkloric dances.**

I. Introduction

Several approaches have been proposed for storing a segment of choreographic data: video recordings of dance performances, choreographer notes, and so on. But how do we approach the issue of storing that segment in a formal fashion, for the purpose of indexing, retrieval, classification, or just for sharing this artistic event with someone else, and in its exact initial form? Dozens of formal languages have been created by dance experts during the last century, but only a few among them are actually used today [2][6][12]. For instance, *Labanotation* provides a standardized way of representing dance in written form. Several attempts have been made to take advantage of the power of computer science (e.g. score editing [16], visualization [9]) in order to improve the way we store dance scores. Until today, data mining techniques have been rarely used for manipulating dance scores [4].

Our research objective is precisely to offer a new approach for dealing with choreographic data. More specifically, we propose an approach that analyzes the choreographic data given in Labanotation by using the notion of "dance tags". In the field of Natural Language Processing, the task of Part-of-Speech (PoS in short) tagging maps each word with its associated PoS (e.g., noun, verb, adjective). By analogy, we propose to map each symbol (or sequence of symbols) extracted from the score with a "dance tag" (e.g., weight shift, sweep, flex). For this purpose, we subject our data to a certain amount of preprocessing and we adapt a rule-based tagger, specifically the one created by E. Brill [5], to our specific needs. This way, our algorithm is able to map the initial choreographic data into a high-level language of basic "gestures". This new comprehensive representation can then be used for applying various data mining techniques: association rules, visualization, classification, etc. In particular, we prove in this paper that our approach leads to promising results in discriminating three genres of dance: classical ballet, modern ballet, and folkloric dances. Our solution might, in future work, be developed into a fully functional framework dedicated to performing such tasks.

The research that has been carried out in this field up to date is rather limited in its range [1][4][8][9][15][17][19]. To the best of our knowledge, the only known previous attempt at machine learning in the field of dance is [18]. This, however, took a different path than the one we have in mind (specifically, motion interpolation). We chose, therefore, to draw inspiration from another work, less elaborate, but closer to our goal, namely a previous attempt [19] to annotate a corpus of dance sequences by manually labeling identifiable motion patterns. We wish to develop an automated version of this process, and then employ a modified part-of-speech tagger to identify "dance tags" and use them to classify the given sequence. As of the date of this work and to the best of our knowledge, ours is the very first such attempt of using data mining and pattern recognition with Labanotation. The repository of Laban scores is very rich (albeit in analogical, not directly usable format) and varied (including classical pieces, but also modern ballet, folkloric dance, etc.). Which is why our solution is a way of valorizing the work of a great number of artists, but it may also lead to several future exciting broader lines of research.

We will begin the main part of this paper (Section 2, specifically) by briefly iterating over our prerequisites in this endeavour and the main previous achievements in the field. We will then, in Section 3, proceed to elaborate on the details of our approach and fully describe the algorithm we use. In Section 4 we will present the experiments we have conducted in order to test our approach, along with the interpretation of our results, and lastly, in Section 5, the conclusions that are to be drawn from these experiments.

II. Related Work

A. *Labanotation*

Along the years, several standardized notations have been suggested for representing dance in written form. The most popular of these (and also the one that we are going to use in developing this work) is the *Laban Notation* (also known as *Labanotation*), which is a component of the wider field of *Laban Movement Analysis* [10][12][13]. This notation allows the description of virtually any type of human motion up to the highest degree of detail and specificity. But at the same time, it does not restrict its usability to a particular type of performance. As such, it may just as well be used in classical ballet, modern dance, and even theater. The actual notation makes use of a set of predetermined symbols in order to precisely represent the direction of the movement, the executing body part, the nature of the motion, as well as the duration of the latter.

B. *Machine-readable representation of Labanotation scores*

The obvious first step in transporting the Laban dance notation to the computer world was to develop a formal representation of a dance score. This representation should capture the semantics of the score in a format that could be automatically interpreted by a machine. The format that we have decided to use in conducting our research is the output format of the LabanWriter application, developed at the Ohio State University [16]. LabanWriter is a software application developed for the Macintosh OS X, that allows users to "draw" from scratch, edit, and store Laban scores in an electronic format. The output file is basically a text file that consists of lines of number sequences, each line denoting an individual symbol or notation on the page.

C. *High-level semantic representation for dance choreography*

As we have previously stated, our main goal in conducting our research is to obtain a high-level representation of dance sequences. We would like this representation to be semantically significant on a machine level, so that any number of advanced procedures (pattern recognition, music recommendation, etc.) can later be carried out in an automated manner.

One promising work in this direction is that of M. Brennan et al. [4]. Their goal was to identify the particular dancer performing an input dance, based on certain peculiarities that were machine "learned" from several dances performed by multiple dancers. Their suggested approach consisted of segmenting the dance sequence at hand into very short fragments that could be labeled with one (or more) of several available tags, from within six categories (such as body actions, shape, etc.). They would then employ a data mining technique to draw a conclusion. In conducting our own research, we plan to maintain the approach of segmentation and tagging, with three key differences, which we will discuss a little further on.

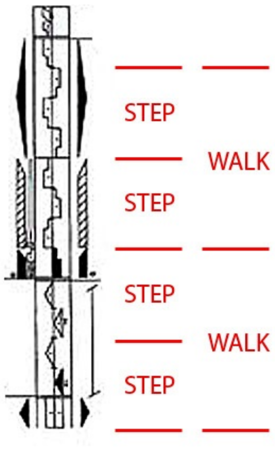


Fig. 1. Hierarchical segmentation of a short movement sequence. The sequence of low-level "step" tokens is subsequently organized into high-level "walk" tokens.

D. *Movement tagging and Part-of-Speech tagging. A parallel between motion and text*

We have mentioned earlier that we are interested in creating a parallel between part-of-speech tagging and "movement tagging", as we have chosen to name the process of segmenting the choreographic information into elementary tokens. The advantages of obtaining a functional analogy such as this are obvious. Part-of-speech (PoS) tagging is already an established field in itself and extensive research has been conducted, with promising and practically usable results [3][7][14]. If we have a working correspondence between the two, we can easily adapt any of several algorithms available for PoS tagging and apply it in our particular scenario.

E. *Segmentation*

When dealing with a corpus that does not hold its semantic load within itself (as text does, for instance), segmentation is in itself quite a difficult step. While it is true that Labanotation scores consist of individual symbols, this segmentation alone is far too low-level to be of much use to us. As such, our first objective in this stage is to divide the score into the shortest movement sequences that are still long enough to be identified as significant and repeatable gestures. Once the initial segmentation is done, we can subject the score to a second segmentation pass, producing higher level entities. Pictured in Fig.1 is an example of a hierarchical segmentation.

On the lowest level we have segmented the sequence into "step" tokens, while on a higher level we group those "step" tokens into "walk" tokens. Each of the two levels provides us with a specific degree of segmentation granularity, and we can later on choose between the two, depending on our particular processing needs.

F. *The Brill tagger approach*

Moving on, we now need to choose a particular Part-of-Speech tagger to adapt to our needs. As far as category is concerned, although statistical taggers generally produce better results, a rule-based approach would be better for us. The main reason for this is the continued performance of the latter even with small datasets (as opposed to the former, which are highly sensitive to this factor). Due to the elaborate and time consuming nature of the translation process between handwritten scores and electronic files (even with the aid of LabanWriter), training data is limited. Our tagger of choice is named the *Brill* tagger [5] and it works in two stages. The first step consists of scanning the entire text and tagging each word with the most likely Part-of-Speech tag, statistically based on the training corpus. The second step consists of a repeated process of acquiring so-called *patches* that aim to improve the output accuracy. A patch is a *rule* by which movements (or "visual words") previously tagged a certain way are retagged based on their context and the trained observation that in this context, the most likely correct tag is another. In Section 3 we will describe the way we plan to adapt the Brill tagger so that we may apply its principles to our problem.

III. Problem Formulation and Suggested Approach

To reiterate the purpose of our project, we aim to create a parallel between Part-of-Speech tagging for text and "dance tagging" for choreographic data. Since PoS tagging is a thoroughly studied field, the creation of this parallel will provide several new leads toward improving the accuracy and flexibility of the type of processing we have expressed our interest in. In order to define the correlation we wish to establish between the two, we must first define the concept of *motion tagging* for dance notation. As we know, a choreographic sequence is composed of a multitude of small individual motions that are to be carried out by various limbs. The Laban notation, as we have seen before, consists of a sequence of special symbols that describe each of these small individual motions or the key body positions between which they evolve. Continuing our parallel with the field of natural language processing, these symbols would be the equivalent of letters. While significant in themselves, they are far too low-level to provide any real, usable information. For us, the process of motion tagging has the purpose of grouping several such adjacent symbols together into a meaningful, identifiable pattern, such as that of a walk, for instance. The equivalent of this would be identifying a group of letters as a specific word. And the final and most important step, the actual tagging, is, of course, the equivalent of determining the part of speech associated with a previously identified word. Once we obtain a tagged dance score, as we have stated before, we can apply any of a number of data mining techniques (based on a large enough corpus of training data that has been tagged manually by a human expert) in order to automatically classify the score or perform a multitude of high-level tasks. The tagging process itself, as well, is carried out automatically, by a pre-trained system. In this section we will now proceed to describe the problem at hand, the structure of our data and our suggested approach.

A. Initial data

Our input data consists of LabanWriter output files, which are essentially text files composed of *lines* of numbers. Each of these lines represents either a part of the description of the staves that compose the score, or, more importantly, one (and only one) of the symbols used in Labanotation. Also, each number on a given line denotes one specific attribute of the symbol or staff description. One important thing to mention here is that due to the overwhelming complexity of the Laban notation, for the purpose of our study we have chosen to consider only a fraction of its capabilities. Specifically, we only consider the most basic and statistically most frequently used symbols (roughly 25% of its capabilities, in terms of symbols and symbol combinations we are able to "recognize"). In future development we aim to include more, and eventually all, of the capabilities of this language.

B. Preprocessing

Our suggested approach consists of two stages, that we are going to describe in detail. The first of these - the preprocessing stage - comprises two subphases of its own. The first of these deals with the parsing of the input file and the conversion of each file line into its *Symbol* counterpart. Basically, the Symbol is a data type that groups together in a stand-alone entity the information provided in the corresponding line that is relevant to our system. The second sub-phase, which we will from here on out call the *first pass*, deals with grouping the primitive Symbols into intermediate level Gestures (we will define the term a little further on). It is worth noting here the difference between a Symbol (capitalized), representing the entity described above, and a symbol (non-capitalized), representing any of the individual shapes that populate the Laban manuscript. Structurally, a Symbol is essentially a tuple of the 5 basic attributes of the motion "token" considered - position in timeline, duration, limb, horizontal orientation, vertical orientation.

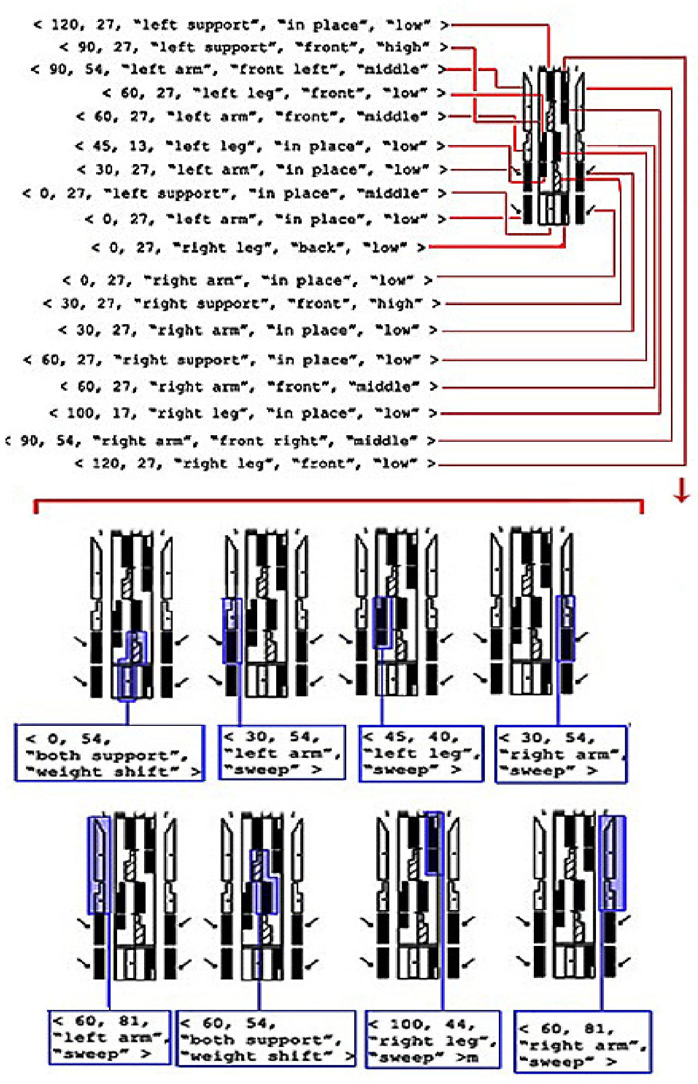
As we have stated before, each of the Symbols that we have extracted directly from the input score in the first preprocessing stage relates to the smallest available division of motion. As such, for the purpose of obtaining better results in the automatic high-level tagging step, we submit the sequence of Symbols that we have obtained in the first preprocessing to an intermediate tagging, the first pass, which is, essentially, a second preprocessing. It is very important to note here that although this step is rule-based, using a small set of rules designed by hand, it is in no way a "recipe" for any high-level tagging. Therefore, it does not interfere in any way with the automatic nature of our tagging system. This stage merely represents a part of the preprocessing phase, and the only reason why we treat it separately from the initial preprocessing is that, unlike the latter, this stage does not map each single token to another single superior token, but rather groups pairs of adjacent tokens together, resulting in a superior level token. More precisely, while the actual preprocessing stage converts score symbols to Symbol entities on a one-to-one correspondence, the first pass converts Symbols to *Gestures* on a two-to-one correspondence.

The way this stage works is by associating a pair of consecutive same-limb Symbols that define a meaningful sequence (such as a jump, or a weight shift, for instance), with the corresponding Gesture. It goes to say that not all sequences of two adjacent Symbols produce a meaningful Gesture. Structurally, a Gesture is basically a tuple of four attributes: position in timeline, duration, limb and gesture type.

We consider a number of six limb channels between which we divide the score symbols, Symbol entities, and Gestures - left and right support (where we define support as a leg with weight on it), left and right leg (no weight on it), left and right arm. During the first pass, we introduce an additional channel - *both support* - where we include the Symbols and Gestures from both regular support channels. The reason for this is that one particular Gesture type is defined by conditions pertaining to both regular support columns, and this approach leads to a simpler decision process.

The Gesture "vocabulary" we have chosen to use comprises the notions of **weight shift**, **sweep**, **weightlessness**, **flex**, and **extend**. For example, a succession of a "left arm low-front" symbol (non-capitalized) and a "left arm high-left" symbol would be identified as a **sweep** Gesture, or a succession of "both legs support" symbols separated by a long enough "break" with nothing present in the support channels is identified as a "weightlessness" Gesture, and so on. Pictured in Fig. 2 is an example of the preprocessing stage, namely the transformation of a set of Symbols (pictured in the upper part of the figure) into a set of Gestures (pictured in the lower part).

One final important observation that we need to make at this point is that unlike the Symbols obtained in preprocessing, which are strictly sequential, the Gestures produced by the first pass of processing may overlap in the dance timeline. The reason we have chosen this approach is to allow the final processing stage greater freedom in selecting the tag that is to be attributed to a given sequence.



< 120, 27, "left support", "in place", "low" >
< 90, 27, "left support", "front", "high" >
< 90, 54, "left arm", "front left", "middle" >
< 60, 27, "left leg", "front", "low" >
< 60, 27, "left arm", "front", "middle" >
< 45, 13, "left leg", "in place", "low" >
< 30, 27, "left arm", "in place", "low" >
< 0, 27, "left support", "in place", "middle" >
< 0, 27, "left arm", "in place", "low" >
< 0, 27, "right leg", "back", "low" >
< 0, 27, "right arm", "in place", "low" >
< 30, 27, "right support", "front", "high" >
< 30, 27, "right arm", "in place", "low" >
< 60, 27, "right support", "in place", "low" >
< 60, 27, "right arm", "front", "middle" >
< 100, 17, "right leg", "in place", "low" >
< 90, 54, "right arm", "front right", "middle" >
< 120, 27, "right leg", "front", "low" >
< 0, 54, "both support", "weight shift" >
< 30, 54, "left arm", "sweep" >
< 45, 40, "left leg", "sweep" >
< 30, 54, "right arm", "sweep" >
< 60, 81, "left arm", "sweep" >
< 60, 54, "both support", "weight shift" >
< 100, 44, "right leg", "sweep" >m
< 60, 81, "right arm", "sweep" >

Fig. 2. Preprocessing stage example

C. Tagging

The actual tagging, or second pass, is the final stage in our suggested approach to dance tagging, and it encompasses the actual creation of the high-level structure that constitutes the main objective of our work. In one of the previous sections we briefly described a Part-of-Speech tagger named the Brill tagger, and stated our desire to adapt its main ideas to our problem and apply its principles in the field of choreographic notation, as opposed to actual text. To reiterate, our proposed approach consists of two steps - an initial statistical tagging, and a final, *patched* tagging. The tag vocabulary that we have chosen to use is as follows: vertical shift (any motion of the body's center of gravity along the vertical axis), walk (a succession of several steps), leap (a jump from one leg to either the other leg, or both legs), hop (also a jump, but from both legs to either one leg, or again both), gesture (a sequence of fluid motion carried out by the same limb), step (as the name suggests, an isolated step), and weightlessness (lack of weight on a leg).

Let us now take a look at the two individual tagging stages.

1) Statistical tagging: From here on, we are going to use the notation *Tag* (capitalized) to denote the label attributed to a given Gesture. As with the Brill tagger, our approach includes an initial tagging phase based solely on a statistical training. What this means is that our system is initially subjected to a training phase, where it is provided with a corpus of sequences that have been correctly tagged by a human. Based on this corpus, our system computes a statistic of what the most likely Tag is for each Gesture. Considering that we use a relatively small vocabulary of Gestures compared to the corpus volume, it is virtually impossible for us to encounter a Gesture with unknown statistical tagging status, i.e. for which we do not have enough occurrences within the training data to compile a reliable statistic. In this stage of tagging we do not consider the context of the Gestures to be tagged, and, as we have stated before, we produce an exact one-to-one mapping.

2) Patched tagging: The second step of tagging, according to the Brill model, is called *patched tagging*. Once all the Gestures have been subjected to the preliminary tagging phase that we have described above, the system reiterates over them and applies all of the eligible patches that it has automatically acquired, therefore modifying temporary Tags to final ones wherever such a modification is required. We will describe the patch acquisition process over the course of the next subsection, but for now we will simply state that, to put it succinctly, patches are contexts in which Gestures previously tagged one way should be retagged in a different way. These patches are acquired fully automatically, without any human intervention.

D. System training

The first part of the training stage consists of elaborating the statistics we have mentioned previously, of the most likely Tag for each Gesture type. For this part, we use a portion of 70% of the available corpus. Another 20% (different from the sample files used in training) we use for patch acquisition, and the remaining 10% is reserved for testing.

For the patch acquisition, we first tag the patch section of the corpus using the statistical training we have just completed. Then we reiterate over the tagged files and compile a list of errors, where an error is viewed as a triplet of an incorrect Tag, the correct Tag that should have been attributed, and the number of times that this exact mistake has been committed throughout the patch training corpus. Then, for each error triplet we determine which of the patch templates and in which context best solves the given error, then we add it to the list of acquired patches.

Listed below are the definitions of the three templates we have designed.

- **template #1**: if t_{old} is preceded by t_a in the same limb channel, then change t_{old} to t_{new}
- **template #2**: if t_{old} is followed by t_a in the same limb channel, then change t_{old} to t_{new}
- **template #3**: if t_{old} is preceded by t_a and followed by t_b in the same limb channel, then change t_{old} to t_{new}

A patch is defined as a tuple of four elements: a patch template, an array of either one or two contextual Tags (as we have seen above, the first two templates require one contextual Tag, while the third requires two), an old Tag (presumed mistaken), and a new Tag (presumed correct, meant to overwrite the initial application of the old Tag). As such, the patch that best resolves a given error is the one that most reduces the number of occurrences of the given error, while introducing as few additional errors as possible (brought on by the cases where, despite the presence of the defining context, the old Tag was actually correct).

E. Pseudocode and schematic representation

Before carrying out any of the tagging steps over a given input file, we need to train the system, subjecting it to a statistical phase and a patch acquisition phase. The pseudo-code for the training stage is presented as Algorithm 1.

Algorithm 1 Training

```
{Input: Training corpus. Output: Final patches list.}
{Statistical training}
for all $file \in trainingCorpus$ do
    for all $taggedGesture \in file$ do
        $tags(gesture, tag) \leftarrow tags(gesture, tag) + 1$
    end for
end for
for all $gesture \in gestureTypes$ do
    $tag(gesture) \leftarrow MostLikelyTag(gesture)$
end for
{Patch acquisition}
for all $gesture \in gestureTypes$ do
    $tag(gesture) \leftarrow MostLikelyTag(gesture)$
end for
for all $taggedGesture \in allFiles$ do
    if $tag(gesture)! = realTag(gesture)$ then
        $error \leftarrow GetError(tag(gesture), realTag(gesture))$
        if $error \in errorsList$ then
            $count(error) \leftarrow count(error) + 1$
        else
            $errorsList \leftarrow errorsList + error$
        end if
    end if
end for
for all $error \in errorsList$ do
    $GetBestPatch(error, suggestedTag, realTag, context)$
    $finalPatches \leftarrow finalPatches + bestPatch$
end for
```

The pseudo-code for the tagging process is presented as Algorithm 2.

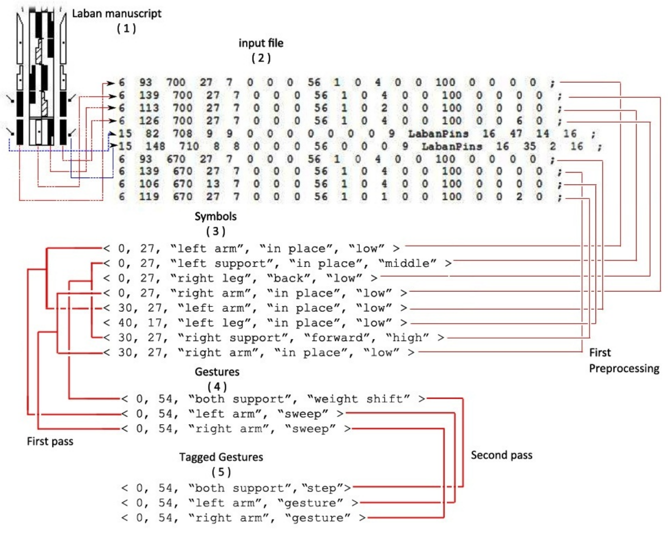
Pictured in Fig. 3 is a schematic of the entire process.Over the course of the next subsections we will proceed to elaborate on each of these steps, explaining them in detail. For now, to briefly iterate over the process, the score file containing lines of numbers defining the most minute symbols are read line per line, and each line is interpreted and transformed into a Symbol (capitalized, representing a data structure comprising all the relevant features of the minute symbol). Symbols are then paired into Gestures (using adjacency and context criteria), and Gestures are then tagged for significance, turning them into Tagged Gestures, as parts of the highest level of segmentation.

IV. Experiments

In order to test the applicability of our high-level framework of dance tagging, we have gathered a dataset of 20 choreographic scores, which we have tagged according to our suggested approach. Then, as proof of concept, we have devised an application that receives a LabanWriter output file as input, tags it according to the algorithm we have described above, and then employs a simple data mining technique (specifically, our own implementation of the Naive Bayes algorithm) to classify it as either a folkloric eastern

Algorithm 2 Tagging

```
{Input: Symbols list. Output: Tagged Gestures list.}
{First pass}
$limbChannelsList \leftarrow GetLimbChannels()$
for all $limbChannel \in limbChannelsList$ do
    for all $Symbol \in limbChannel$ do
        $Gesture \leftarrow GetGesture(Symbol, next(Symbol))$
        $GesturesList \leftarrow GesturesList + Gesture$
    end for
end for
{Statistical tagging}
for all $Gesture \in GesturesList$ do
    $TagGesture(Gesture, MostLikelyTag(Gesture))$
end for
{Patched tagging}
for all $Gesture \in GesturesList$ do
    for all $patch \in patchesList$ do
        if $IsAppliccable(patch, gesture, context)$ then
            $ApplyPatch(Gesture, patch)$ $Retag(Gesture)$
        end if
    end for
end for
```



```
6   93  700  27  7  0  0  0  56  1  0  4  0  0  100  0  0  0  0  ;
6  139  700  27  7  0  0  0  56  1  0  4  0  0  100  0  0  0  0  ;
6  113  700  27  7  0  0  0  56  1  0  2  0  0  100  0  0  0  0  ;
6  126  700  27  7  0  0  0  56  1  0  4  0  0  100  0  0  6  0  ;
15  82  708   9  9  0  0  0   0  0  0  0  9  LabanPins  16  47  14  16  ;
15 148  710   8  8  0  0  0  56  0  0  0  9  LabanPins  16  35  2  16  ;
6   93  670  27  7  0  0  0  56  1  0  4  0  0  100  0  0  0  0  ;
6  139  670  27  7  0  0  0  56  1  0  4  0  0  100  0  0  0  0  ;
6  106  670  13  7  0  0  0  56  1  0  4  0  0  100  0  0  0  0  ;
6  119  670  27  7  0  0  0  56  1  0  1  0  0  100  0  0  2  0  ;
```

```
< 0, 27, "left arm", "in place", "low" >
< 0, 27, "left support", "in place", "middle" >
< 0, 27, "right leg", "back", "low" >
< 0, 27, "right arm", "in place", "low" >
< 30, 27, "left arm", "in place", "low" >
< 40, 17, "left leg", "in place", "low" >
< 30, 27, "right support", "forward", "high" >
< 30, 27, "right arm", "in place", "low" >
```

```
< 0, 54, "both support", "weight shift" >
< 0, 54, "left arm", "sweep" >
< 0, 54, "right arm", "sweep" >
```

```
< 0, 54, "both support", "step">
< 0, 54, "left arm", "gesture" >
< 0, 54, "right arm", "gesture" >
```

Fig. 3. Schematic of the tagging process.

European dance, or a classical ballet piece. We have chosen this particular algorithm (in its bare-bones implementation) because it is one of the most basic ones available, making for a fair test of our framework. Also, without the algorithm optimizations in other implementations, we are guaranteed that the classification process will not skew our results in the segmentation and tagging portion.

A. Dataset

The training stage (statistical and patch acquisition) is performed using a corpus of twenty choreographic pieces, all pre-tagged manually (for algorithm evaluation purposes). We have ten classical ballet pieces and ten folkloric pieces. The scores have an average of 212 primitive symbols (the number of primitive symbols in each file corresponding to the number of useful lines in that file), and acquire an average of 168 Gestures after the preprocessing stage. In other words, the total volume of data that we have used is of approximately 20 * 212 Symbols, or 20 * 168 Gestures.

B. Automatic tagging

Let us now take a look at the actual results we have obtained. Even with statistical tagging alone, the accuracy is of 70 to 80%, and after applying the acquired patches, we obtain an accuracy of over 85%, and as high as 94% (in classical ballet vs. folkloric dances), which is a very promising result. Two examples of actual patches that our system has acquired ("translated" into natural language) are as follows: "If <step> tag is preceded and/or followed by another <step> tag, then change it to <walk> tag" and "If <hop> tag is preceded by <step> tag, then change it to <leap> tag".

Pictured in Fig. 4 are graphics of the accuracy levels we have obtained for different dance scores from the two categories we have considered.

This goes to show that in our particular scenario, the rule-based approach is indeed an improvement over the traditional statistical approach, since, as we have explained before, the former is far more reliable when using a rather limited volume of training data.

C. A potential application - genre classification

As a means of testing our theoretical framework, we have implemented an application that employs our solution to dance tagging (as we have described it over the course of the previous sections) and used it to decide whether a given dance score is either a classical ballet piece, or a folkloric eastern European dance. After the two main stages in our solution (preprocessing, and actual tagging), the application displays the list of tokens produced (be they Symbols, Gestures, or Tags), along with a brief statistic compiled at the end of each stage, and, of course, the final decision in relation to the identified genre.

For the verification part of out experiment, which meant evaluating the classification accuracy when not using our system, we considered the symbols in each score (uncapitalized, meaning the lowest level entities present) to be the features. For the actual testing, we considered the tagged Gestures, context sensitive, as the features. In both cases, the training set consisted of 70% of our entire score collection (meaning seven complete dances, out of the ten we had available for each of the three categories considered - classical ballet, modern ballet, folkloric dance), and the testing set consisted of dance sequences of varying lengths (but no shorter than a quarter of the average length over all the complete dances) extracted from all 100% of the available dances. The reason for imposing this minimal length was that the complete dance sequences in themselves being relatively brief (around four minutes of dance, on average), sequences any shorter than this would

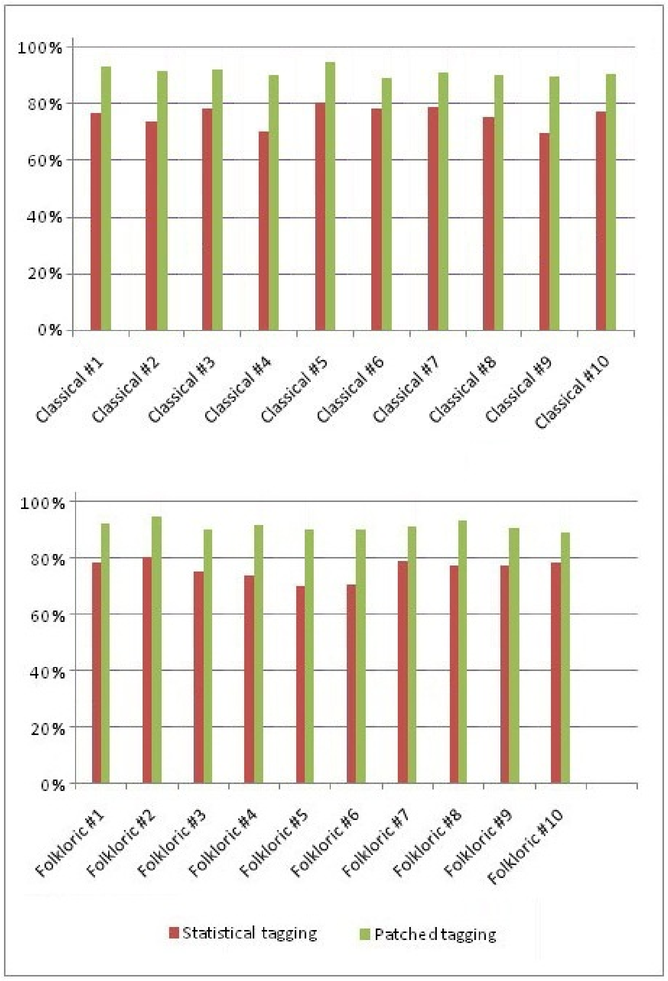


Fig. 4. Accuracy output for classical and folkloric scores (in Gesture tagging). The accuracy for a given dance is the accuracy with which the Gestures composing that dance are correctly tagged (according to their known human-assigned tag).

TABLE I
TEST RESULTS FOR CLASSIFICATION STAGE (FOR TAGGED SCORES, CONTEXT CONSIDERED).

Algorithm	**Classical vs. Folkloric**	**Classical vs. Modern**
Naive Bayes	98%	72%
ID3	97.3%	74%
AdaBoost	98.5%	75%
K-Nearest Neighbor	96%	70.2%
C4.5	99%	72.4%
Random forest	94.3%	69.5%

TABLE II
TEST RESULTS FOR CLASSIFICATION STAGE (FOR UNTAGGED SCORES, CONTEXT NOT CONSIDERED).

Algorithm	**Classical vs. Folkloric**	**Classical vs. Modern**
Naive Bayes	65.3%	51.0%
ID3	70.8%	56.4%
AdaBoost	72.0%	64.8%
K-Nearest Neighbor	59.7%	48.2%
C4.5	75.1%	66.3%
Random forest	68.8%	47.3%

be far too limited for classification to even make sense. For instance, one might imagine trying to classify one generic step as part of a dance genre - the shorter the sequence, the more universally applicable it is.

As far as results are concerned, we have obtained an accuracy of 98% in automatically differentiating between classical ballet and folkloric dance, based on our framework. This is, however, partly due to the fact that the two categories have been specifically selected to be sufficiently different in aspect (as far as general rythm and most frequent pieces of movement are concerned). We have also tested with a smaller number of modern ballet pieces versus classical ballet (the two being much more similar than the classical ballet - folkloric dance pairing), and the accuracy has predictably declined to 72% (which is, however, still quite a promising result). Beside our own implementation of the Naive Bayes classification algorithm, we have also tested our data with several other decision algorithms, using the Weka framework [11]. The parameters used in testing these algorithms were the default ones, since we are aiming to see the baseline result for each of these algorithms, and since different algorithms have different available degrees of optimization. We will not elaborate here on these specific parameters, since this application is merely a proof of concept, and these parameters could be modified in a multitude of ways, leading to potentially better results. What we are interested in is a baseline for each of these algorithms. Listed in TABLE 1 are the results we have obtained, in terms of classification accuracy, when using the fully segmented and tagged scores (by means of our suggested algorithm).

As we can see, the classification accuracy is quite close for all the algorithms tested, proving that our approach is universal, and a good starting point for subsequently applying either of these algorithms, and is not limited to functioning with any single one. Also, the utility of our segmentation and tagging process is proven by the difference between these results and the ones obtained when using the unprocessed scores (in their original, lowest level segmentation, of Laban symbols), listed in TABLE 2. The main reason for this decline in accuracy is the increased occurrence of overfitting when considering each tiny Laban symbol as a relevant feature. This, coupled with the limited amount of training data that would provide examples of all the contexts and symbol combinations that we can expect and their relevance in the classification process, fully explains the decline in results when removing our tagging layer.

As a partial conclusion, we can say that although our dataset was limited (due to the scarcity of available data already in usable electronic format, despite its abundence in analogical format, and thus the necessity for us to convert our own data, which is a highly time-consuming process) we have been able to prove that our suggested approach is not only feasible, but also able to produce very promising results, considering our achieved accuracy, both in tagging and in genre classification.

V. CONCLUSIONS AND OUTLOOK

In order to reiterate our proposed approach, our main goal (and original contribution to this field) was to design a machine learning algorithm based on rules and inspired by the main concept of the Brill Part-of-Speech tagger, dedicated to enabling the performance of several high-level tasks, such as pattern recognition, music recommendation, etc. We began with an input file in text format, consisting of a large array of lines of numbers. This input file encodes a Laban score, and each of its composing lines of numbers corresponds to a particular Laban symbol. In the preprocessing stage, we first parse the file and group the information in each line of text into a Symbol entity, which is the smallest meaningful entity that we can work with in our application. Having done that, still in the preprocessing phase, we group pairs of consecutive Symbols into intermediate-level entities named Gestures. We then move on to the actual tagging, which, in turn, has two parts of its own. In the first part, we tag each Gesture based on its most likely association in the training corpus, while in the second part we determine a set of patches that improve tagging accuracy in the training corpus. Once the system has been trained and a file has been subjected to the tagging process, we can then employ any of a large number of data mining techniques in order to decide the genre of the dance, based on its annotation. Our solution brings innovation to the fields of Machine Learning and Pattern Recognition and paves the way to several possibilities as far as potential applications are concerned. Not only did we expand the field of automated pattern recognition to the dance world, but our approach has, as we have seen, yielded very promising results, both in the tagging stage and in the classification stage. There is, however, much more that can be done in future work. For the sake of simplicity and a more focused evaluation of our suggested approach, we have considered a greatly reduced portion of the Labanotation vocabulary. Before this solution can be made available to the general public and commercially used, we need to scale it up to include the entire array of Laban symbols and notations. There is still much work to be done in order for our approach to accommodate the entire richness of this complex language. One favorable thing about this direction, however, is that we have implemented our solution in such a way as to easily allow the expansion of the vocabulary with which we have designed the system to work. Once we expand our system to “understand” a wider array of notations and symbols, we can also expect our classification accuracy to improve as well. The reason for this would be the newly acquired access to a greater number of criteria by which to judge the closeness of a sample dance sequence to one genre or another.

And lastly, one other potential direction of development for our work would be to provide our system with an interface enabling input in motion capture form. As such, a given dance sequence could be entered for processing either by means of Labanotation scores (as it is now), or by direct motion capture from a dancer performing the given sequence of movements. The advantage of this latter approach is mainly the improved precision in describing the movement we wish to analyze, as well as the smaller amount of time and effort required for obtaining our input. As we have stated before, although Laban scores are widely available, very few of these can be found in an electronic format that is machine-readable. As such, at the moment, a considerable amount of manpower needs to be employed in order to manually convert handwritten Laban scores to LabanWriter files (or any other file format that we can work with in an automated manner). By representing movement as data acquired with the aid of motion capture equipment, we could circumvent this entire process and have quicker access to a much larger dataset.

REFERENCES

[1] N.I. Badler and S.W. Smoliar, *Digital Representations of Human Movement*, University of Pennsylvania, Pennsylvania, 1979.

[2] R. Benesh and J. Benesh, *Reading Dance: The Birth of Choreology*, McGraw-Hill Book Company Ltd, 1983.

[3] T. Brants, *TnT: A Statistical Part-of-Speech Tagger*, Proceedings of the sixth conference on Applied natural language processing, 2000.

[4] M. Brennan, *A Computerized Methodology for Recording and Analyzing Movement Qualities*, Dance Notation Journal, 1986.

[5] E. Brill, *A Simple Rule-Based Part of Speech Tagger*, University of Pennsylvania, Pennsylvania, USA, 1992.

[6] N. Eschkol and A. Wachman, *Movement notation*, Weidenfeld and Nicolson, London, 1958.

[7] S. Goldwater and T. Griffiths, *A Fully Bayesian Approach to Unsupervised Part-of-Speech Tagging*, The Annual Meeting of the Association for Computational Linguistics, 2007.

[8] K. Hachimura, *Digital Archiving of Dancing*, Ritsumeikan University, Kusatsu, Japan, 2006.

[9] K. Hachimura, K. Kojima and M. Nakamura, *Laban Editor: Graphical Editor for Dance Notation*, Proceedings of the 2002 IEEE Int. Workshop on Robot and Human Interactive Communication, Berlin, Germany, 2002.

[10] D. Herbison-Evans, *Dance and the Computer: A Potential for Graphic Synergy*, University of Sydney, 2003.

[11] G. Holmes, B. Phahringer, P. Reutemann, M. Hall, E. Frank and I.H. Witten, *The WEKA Data Mining Software: An Update*, ACM SIGKDD Explorations Newsletter, vol. 11, nr. 1, p. 10-18, 2009.

[12] A. Hutchinson and A.H. Guest, *Labanotation, or Kinetography Laban: The System of Analyzing and Recording Movement*, Dance Books, 1996.

[13] G.P. Kurath, *The Journal of American Folklore*, JSTOR, 1957.

[14] A. McCallum, J. Lafferty and F. Pereira, *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*, Machine Learning - International Workshop and Conference, 2001.

[15] M. Nakamura, W. Choi, W. Choensawat, S. Takahashi and K. Hachimura, *Description and Reproduction of Stylized Traditional Dance Body Motion by Using Labanotation*, Transactions of the Virtual Reality Society of Japan, Vol.15, No.3, p.379-388, 2010.

[16] L. Ross, L. Venable, S. Sutherland and M. Tinsley, *Laban-Writer 2.0*, The Ohio State University, Department of Dance, 1989.

[17] R. Ryman, T. Calvert, W. Wilke and I. Fox, *Applications of Computers to Dance*, Computer Graphics and Applications, IEEE, vol. 25, nr. 2, p. 6-12, 2005.

[18] J. M. Stuart, and E. Bradley, *Learning the Grammar of Dance*, Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998

[19] A. Trexler, R.K. Thornton, M. Apostolos, D. Petty, N. Badler, T.W. Calvert, S.D. Kahn, R. Rhyman, P. Dozzi, J.A. Gray, M.A. Brennan and G. Politis, *Dance Technology: Current Applications and Future Trends*, The American Alliance for Health, Physical Education, Recreation, and Dance, Virginia, USA, 1989.