

Exploring Automatic Recognition of Labanotation Dance Scores

Michelle de Böck

Supervisor: V. de Boer

Department of Sciences
Master Information Sciences
Vrije Universiteit Amsterdam

27-05-2019

Abstract

Digitalization helps to archive and protect cultural heritage, including as dance, music and arts. However, there still is a limited amount of cultural preservation devoted to performing arts, more specifically to recording and capturing dance. Rudolf von Laban developed in 1920 a written dance notation style called Labanotation, which makes use of symbols to record human movements in a well-structured format. The problem with Labanotation is that its complex and difficult to understand. Therefore, we develop a new method that can read and interpret Labanotation image files. More specifically, we investigate developing an Optical Labanotation Recognition (OLR) Tool that is able to interpret the Labanotation files in MATLAB. For the sake of clarity, this research is divided into two separate parts, named Part I and Part II. Part I of the research focuses on recognizing individual Labanotation elements at a time on three different machine learning algorithms that are considered to be effective in finding the design features of an Optical Labanotation Recognition (OLR) Tool. Such a tool can assist choreographers in creating new dances, but also stimulate dancers in their creativity. The results of Part I show that deep learning techniques with AlexNet scores the highest accuracy. In Part II of this research we investigate a Multi-Laban detector, which is evaluated by two original handwritten Labanotation files, that serve as test cases. The results of Part II show that the accuracy is low when using an original handwritten Labanotation file, mostly due to challenges such as noise and blur that are present during the digitalization process. Based on these results, we conclude this study by proposing design features for an Optical Labanotation Recognition Tool.

1 Introduction

1.1 The problem: capturing dance on paper

The process of archiving and protecting cultural heritage has seen a large transformation, due to the advent of new technologies in digitalization as well as digital reconstruction (Tsirliganis et al., 2004). These transformations include the way information is being retrieved, stored and presented. Examples of cultural heritage are monuments, archaeological buildings or ancient works of art (Blake, 2000). Preserving cultural objects digitally is important for securing historical data, archiving purposes and making these cultural objects available at anytime from anywhere (Vilbrandt et al., 2004). Other examples of cultural heritage include performing arts, such as dance, music and theatre (Wikipedia contributors, 2018). Performing arts are present in all human cultures and can be invested more in regarding the digitalization of such cultural heritage.

Especially in the area of dance, there is a limited amount of cultural preservation devoted to performing arts, more specifically to recording and capturing dance styles (El Raheb & Ioannidis, 2012). According to Johnson & Snyder (1999), a predominant reason of this limited amount of research devoted to dance lies historically within the problem of capturing the movement within the dances. Dance is usually captured with recording equipment, whereas capturing music notes, paintings or photographs are captured best in two dimensions and have commonly understood storage mechanisms today. In this thesis, the problem of capturing and understanding dance in a two-dimensional environment is being addressed by elaborating on the written dance language called Labanotation as explained further in Section 1.3.

1.2 Different techniques to capture dance

Dance can be captured and archived through one of the following capture methods: video recording, motion capture, and a written dance notation of which Labanotation is the most generally used notation format, for depicting body motion (Nakamura & Hachimura, 2006). Video recording can capture and record dance in three-dimensional animation. However, video recording alone cannot analyze the movements of the dance style and cannot be used for programming purposes, unless the movements are modelled and expressed in mark-up languages (Chung et al., 2005).

The second technique, which is motion capture, allows motion data to be created with physical movement and is commonly used in sports and entertainment (Noonan et al., 2009). The motion capture data is then mapped onto a three-dimensional model, performing the same movements as the actor or object does. However, this technique is relatively expensive, requires specialized labor, and moreover, is time-consuming (Chung et al., 2005).

The last of the three techniques mentioned, Labanotation, is a written dance notation that first was introduced in 1920 by the architect, theorist, and choreographer named Rudolf von Laban (El Raheb & Ioannidis, 2012). Labanotation is a written language that uses symbols to record human movements in a well-structured format, with clear semantics to describe the domain of the dance movement (El Raheb & Ioannidis, 2012).

Hat (2006) argued that Labanotation is flexible enough to represent any type of movement. Labanotation has been widely used in America, Europe and in Asia (Tongpaen et al., 2017). This written language is used as a general body motion notation, and does not depend on a specific dance style (Nakamura & Hachimura, 2006). Figure 1 depicts the creator of Labanotation, where he is presenting his notation system. The notation is based on graphical representation, that is illustrated in Figure 2. Section 2 will elaborate further on the details of describing Labanotation symbols.

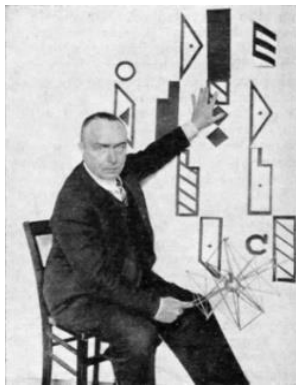


Figure. 1 The creator of Labanotation named Rudolf von Laban (Aventina, 1928)

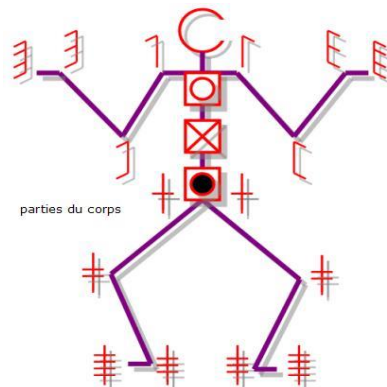


Figure 2. Symbols referring for parts of the body (Huster, 2007).

1.3 Research Gap

Labanotation is a written human movement language used for notating dance (Barbacci, 2002). This notation is a standard graphical representation that is available for recording dance movements and choreographies. In contrast to a musical notation, examples of Labanotation documents are not so prevalent and often not digitally available. According to the Dance Notation Library there are over 600 original Labanotation documents achieved, whereas there are over 97 million songs currently present (King, 2011). The 600 original Labanotation documents presented in the Dance Notation Library cannot be download for free. The Labanotation documents that are free are in many cases a photocopied scan of paper documents and not directly machine-readable, in a sense that the symbols cannot be automatically recognized and be distinguished from one another.

In this research we are going to address this problem by developing a method that is able to read existing as well as already computer-generated Labanotation dance documents. More specifically, we develop a new technique called Optical Labanotation Recognition (OLR) that is able to interpret and understand the Labanotation files automatically in a software tool called MATLAB. The OLR technique that is utilized will recognize and classify the meaning of the scanned chorographical samples. This paper will not investigate how the OLR technique reads the Labanotation files and convert it into a 3D animation. The research in this paper will in fact investigate the gap of reading the scanned Labanotation automatically and classify the correct symbols with the help

of a MATLAB script. Based on this research, the following research question can be answered:

Research Question: What are the design features of an Optical Character Recognition program that can read scanned Labanotation documents?

The scope of this research is focused on recognizing the most used Labanotation symbols. The design features will be determined based on the performance of the three machine learning algorithms executed in the software tool MATLAB (Section 4).

According to de Boer et al. (2018), the major challenges lie in the fact that most choreographed dance routines are not usually stored in a retrievable form that could be used in a later stage in programmable algorithms. Dance movements are either stored in video recordings as mentioned previously or are simply retained in the memory of the choreographer. The absence of a valuable representation of dance movement and choreography in retrievable form has hindered the role of computers to simulate the innovative and creative processes in dance (de Boer et al., 2018). According to Calvert et al. (2005), the creation of traditional choreographies is rather cost intensive as well as time-consuming. Using accurate computer software could reduce the cost and time needed, allowing for the process to be a lot more efficient (de Boer et al., 2018). Another present challenge about denoting dance movement is the lack of creativity in making new dance choreographies (Dyke, 2001).

This paper contributes to the existing literature on Labanotation by developing a new technique called Optical Labanotation Recognition (OLR) that is able to interpret Labanotation scores. In the future, the OLR is able to create new options for choreographies or for more variety in the steps, addressing this creativity challenge that is present. In conclusion, the atomization of interpreting Labanotation documents by computer software can be considered as the first step towards creating a so-called dance-generator.

2 Background Labanotation

2.1 Introduction of Labanotation

Labanotation represents a number body parts in nine distinct columns, also called a staff (Sankhla et al., 2018). Figure 3 below shows the examples of the notation scores as follows: (a) each staff is a part of the body, for example the left leg is positioned on the column number two on the left side, same for the right side only for the right leg; (b) is the structure: the vertical axis refers to time. Each horizontal line represents a space that is similar to the bar lines in music notes, i.e. the music notes A, B, C; (c) the direction: each symbol shows the direction of the movement (Sankhla et al., 2018).

Figure 4 below describes the level of the direction that is represented by a texture. The texture of the Labanotation symbol indicates the level of movement. Striped has a high level, which means up; with a dot in the center has a middle level, which means parallel to the floor; black has a low level, which means down (Barbacci, 2002). The

duration of the movement is represented by its length. The shorter the symbol, the shorter the duration of the represented dance movement. The longer the symbol, the slower the dance movement is. To represent a movement of a specific body part, the Labanotation author would put the appropriate symbol on a column of the staff (Nakamura & Hachimura, 2006).

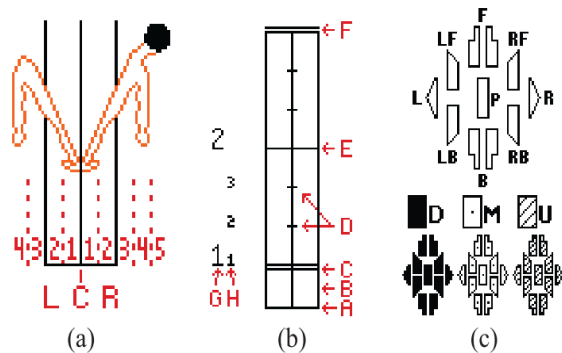


Figure 3. Explanation of Labanotation scores (Sankhla et al., 2018)

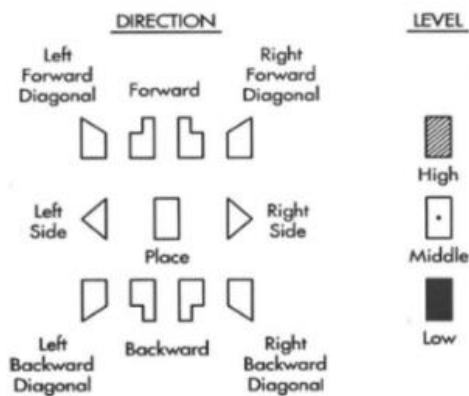


Figure 4. Direction of Labanotation scores (Nakamura & Hachimura, 2006)

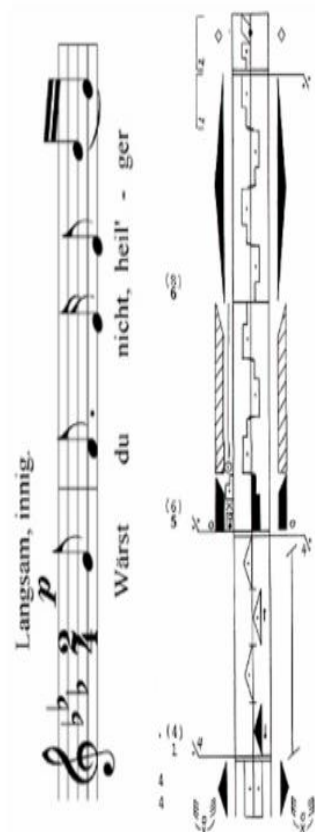


Figure 5. Music notes (left side) with the corresponding Labanotation scores (right side) (Nakamura & Hachimura, 2006)

Labanotation language is difficult to read and even more difficult to write as described by El Raheb & Ioannidis (2012). For clarity, Figure 5 above illustrates on the left side the music notes and pm the right side the Labanotation symbols. Contrasting from most western languages which use horizontal notation, the notation for Labanotation is vertical and reads from bottom to top. The directionality combined with a wide variety of symbols available within the notation results in notation which many people find enormously complex and difficult to understand (Baloh et al. 2015).

2.2. Software programs with Labanotation

There exist graphical editors that are able to create Labanotation documents in a software program. The well-most known graphical Laban processing software are: LabanWriter, LabanDancer, LabanXML and LabanEditor, which we will describe below.

The first most used graphical editor called LabanWriter (Calvert et al., 2005), and provides the user with a blank canvas on where the user can create its own Labanotation using of various symbols. An advantage of LabanWriter is that the Labanotation score either can be printed, but also be exported as a 2D raster image in the following three formats: .png, .pict, or .jpeg. (Calvert et al., 2005).

Another Laban processing software is LabanDancer. LabanDancer translate the written Labanotation files into 3D human figure animations. This is a major advantage due to the fact that most choreographers and dancers do not have adequate knowledge in writing and/or reading written Labanotation language (Wilke et al., 2005).

The next Labanotation software that we introduce is LabanXML, which is an XML styled representation of Labanotation developed by Nakamura & Hachimura (2006). The structure of LabanXML exists of a root element, also called the `<laban>` element. The `<laban>` element includes the following elements: `<attribute>` and `<notation>`. The `<attribute>` element includes `<time>` element. Furthermore, the `<time>` element includes a `<beat>` and a `<beat-type>` element. Appendix Section 10.1 illustrates a short code of LabanXML.

LabanEditor has been developed by the Ritsumeikan University and is an interactive graphical editor for editing and writing Labanotation scores (Nakamura & Hachimura, 2006). The benefit of LabanEditor is that the user has the possibility to edit dance movements and display the Labanotation score via an animation of human body models in 3D graphical notation (Hat, 2006; Nakamura, 2006). Research papers have stated that the LabanEditor can also read and write LabanXML, however, this software no longer available by it's publisher (Baloh et al., 2015).

Although it would be mostly valuable to use these editors in our research, LabanXML and LabanEditor no longer exist, and simply cannot be found on the Internet anymore (Baloh et al., 2015). LabanWriter is only supported on MacOS and LabanDancer is still available, but has no added value since our research does not focus interpreting Labanotation documents into 3D human figure animations. Nevertheless, this research does not depend on these Labanotation software tools, since it instead will investigate a new developed method called Optical Labanotation Recognition (OLR), that will investigate on how to read and interpret scanned Labanotation files. The OLR technique is inspired by the Optical Character Recognition (OCR) method as described in Section 3.1.

3 Related Literature

3.1 Optical Recognition Techniques

Optical Character Recognition (OCR) is a technique that is able to classify optical patterns that correspond to individual letters, numbers or other characters, either single letters or multiple combined in a word or sentence (Hansen, 2002). The OCR process exists of several steps: segmentation, feature extraction, and classification. OCR has developed into one of the most successful applications of pattern recognition (Tiwari et al., 2013). The benefit of OCR is that it relates a symbolic identity with that of the image of a character (Tiwari et al., 2013).

Optical Character Recognition technology can be implemented into MATLAB. MATLAB is a programming software environment designed specifically for scientists and engineers and uses computational mathematics (MathWorks, 2019b). Using OCR in MATLAB provides, for instance, results in the field of word detection and recognition of natural images (Wang et al., 2011). Wang et al. (2011) created a baseline for using OCR techniques for end-to-end word recognition for real-life use cases within MATLAB. Furthermore, another study conducted by Aher & Kapale (2017) investigated an automatic number plate recognition system for vehicles using OCR technology in MATLAB (Aher & Kapale, 2017).

Besides OCR, there is also the possibility to classify music notes using a method called Optical Music Recognition (OMR) (Rebelo et al., 2010; Rebelo et al., 2012). However, the challenge Rebelo et al. (2010) faced was recognizing handwritten music recognition and solving it by transforming the paper-based music manuscripts into a machine-editable symbolic format, that it can be used with the OMR algorithm. Rebelo et al., (2012) described the following advantages of using an OMR system: (1) automated transformation of paper-based music manuscripts into machine-readable format, thus saving time, (2) implementing translations such as Braille notations, and (3) preservation of cultural heritage. Given all these points, in this thesis we use a derivative OCR method, but with the goal to recognize Labanotation symbols in a scanned document (Section 4).

3.2 Other classification problems in MATLAB

Netzer et al. (2011) addresses the problem of recognizing digits using unsupervised feature learning methods. They developed an application that can read house numbers from photos of the street. Interesting in their research was that the performance increased rapidly for all the executed methods, when they increased the training examples up to 100.000 (Netzer et al., 2011). Another classification problem in Object Detection is to detect humans in images (Dalal & Triggs, 2005). Dalal & Triggs (2005) addressed the object detection problem by normalized Histogram of Oriented Gradient (HOG) method.

The study of Dodge & Karam (2016) argued the importance of have sufficient data quality as input for machine learning purposes. Despite having images that are resilient

to compression artifacts and contrast, they concluded that all neural networks are susceptible to blur and noise. This an interesting finding because it implies that the reduced performance under blur and noise is common to the considered Neural Network classification models (Dodge & Karam, 2016).

In our research paper we face challenges such as having a low number of training samples and will be elaborated on more in Section 4.2.

3.3 Pretrained Neural Networks

One of the common approaches for image classification is to train a classifier model from scratch. Unfortunately, designing a classifier from scratch is only recommended when the training data consists of 1000s to millions of labeled images (MathWorks, 2019). When having small datasets there is a high risk of overfitting. Another downside of training a classifier model from scratch is that it requires intensive computation and lots of time to train.

A solid solution to deal with the intensive computation time and training time, is to use pretrained neural networks. Pretrained neural networks are already trained on over more than a million images from the ImageNet database (ImageNet, 2019), and are able to classify images into 1000 object categories, such as: animals, keyboards and pencils. Using pretrained neural networks reduces training time and are easy to use. There are a number of pretrained neural networks available: AlexNet, which has a neural network of eight layers deep (Krizhevsky et al., 2012, Russakovsky et al., 2015); GoogleNet, which has a neural network of 22 layers deep (Szegedy et al., 2014); and ResNet-50 has a neural network of 50 layers deep (He et al., 2015). Rio et al., (2018) stated that ResNets are often the best feature extractors in image classification, because they do not depend of their ImageNet database accuracies. In our research of Part I, we used the pretrained neural network AlexNet in two machine learning algorithms, since it reduces training time.

4 Method

4.1 Engineering Study

To investigate the design features of an Optical Labanotation Recognition Tool, we perform an engineering study, using iterative design, to investigate individual Laban symbols first. In Part I of the research, we investigate three different machine learning algorithms in order to identify individual Laban symbols. When the engineering stage of Part I is completed, we continue to the evaluation stage, where we test and evaluate the algorithms based on their accuracy, recall and precision. The aim of this research in Part I is to provide conceptual design features of the Optical Labanotation Recognition Tool. All algorithms were developed in MATLAB.

























4.2 Prerequisite: creation of Image Database

In order to use machine learning algorithms for categorizing individual Laban symbols, first we need a Labanotation dataset that could tell us what each individual Laban symbol means. As no pre-labeled dataset of Laban symbols is available, our first step is to create such dataset for machine learning purposes. This dataset is used both in Part I and Part II of this research. Having a pre-labeled dataset would dramatically reduce the time needed for creating a dataset, resulting in increased training time allowance of each algorithm, meaning the algorithms could become more sophisticated.

Sufficient quality of the self-created Laban database is essential, as the quality of the algorithms solely depends on the data input quality in machine learning algorithms (Dodge & Karam, 2016). Therefore, the best suitable approach for creating an image database, is to label each independent Laban symbol manually, attributing them to their respective category. The advantages of doing so, makes that all categories will have the same number of images, so that there are no unbalanced weighted categories. In the case of an unbalanced dataset, the classification model becomes more biased towards the majority class category, as it has a larger influence on the final classification outcome and the model becomes less valuable (Maheswari, 2018).

The Laban image dataset is created in a graphical editor called LabaNotator and are stored in a .jpg file format. LabaNotator is a software tool which enables the possibility to edit existing Labanotation files and also has support for adding new Laban symbols. On top of that, the availability of existing pre-defined symbol libraries helps in writing and editing these Laban files (Bezjak, 2014). The Laban image dataset is used as input for all three machine learning algorithms and is illustrated below in Table 1. For research purposes and more details, anyone can download the dataset from GitHub at: <https://github.com/michelledebock/Labanotation>.

Table 1. Accuracy obtained from the algorithms to identify Laban symbols (Bezjak, 2014.)

ID	Category	N=1	N=2	N=3	N=4	N=5	N=20	Description
L1	Left Forward							Left Forward High
L2	Left Forward							Left Forward Low
L3	Left Forward							Left Forward Normal
L4	Place							Place High

L5	Place							Place Low
L6	Place							Place Normal
L7	Right Forward							Right Forward High
L8	Right Forward							Right Forward Low
L9	Right Forward							Right Forward Normal
L10	Direction Left							Direction Left High
L11	Direction Left							Direction Left Normal
L12	Direction Left							Direction Left Low

4.3 Training set and Test set

Table 3 represents a subset of the total Labanotation image database, which consists of four categories namely: ‘Left Forward’, ‘Right Forward’, ‘Place’, and ‘Left Direction’. The four aforementioned symbols are the most elemental steps when initiating a Labanotation document (Fügedi, 2006). Every category in its turn has three distinct symbols, each with a different padding, (either black, striped or dot in center). The total Labanotation image database consists of 60 images per category, thus a total of 240 images being the entire Laban image dataset.

For machine learning purposes, and for each class, we randomly split the available image dataset into a 70% training set and a 30% test set. This is particularly important, due to the fact that the algorithm is learnt on the training set and hereafter the classifier is validated on the test set. This is to prevent the problem of ‘overfitting’, that would otherwise occur. The algorithm learns on the whole data set, and tests itself on the same dataset continuously, and therefore is not able to detect any Laban symbols anymore (Hariharan, 2018).

4.4 Engineering Stage

Part I of this research is about investigating individual Laban symbols and classify them according to three distinct machine learning algorithms: (1) Object Recognition using machine learning algorithms (Csurka et al., 2004), (2) Object Recognition with Transfer Learning (Krizhevsky et al., 2012), (3) Object Recognition with Feature Extraction Using AlexNet. These three algorithms are used for classifying images or objects, but never used for classifying Labanotation symbols. Hence, we aim on using the algorithms as a basis from where we construct our own algorithms, derived from them, which are best suitable for classifying Laban symbols.

The main goal of the engineering stage is to train the Optical Laban Recognition Tool in such a way that it can predict the appropriate category of a not before seen Laban symbol. In the following sections, the functionality for each algorithm is explained in means of pseudocode, supported by a detailed description. Providing the pseudocode version of each algorithm helps people to understand the step-by-step actions that must be taken (Pogue, 2019). For research purposes and more details, anyone can download the algorithms from GitHub at: <https://github.com/michelledebock/Labanotation>.

4.4.1 Object Recognition with Ensemble Bagged Trees Classification

Figure 6 below represents the pseudocode of this algorithm which is described as follows:

1. The function *Visual Bag of Words* is used to extract features from these Laban symbol images. *Visual Bag of Words* represents each Laban symbol image in words and counts the occurrence of these 'visual words'. To represent this in a visualization, the function then generates a histogram. Based on the word frequencies of the histogram, the images can each be classified. In addition, the function *Visual Bag of Feature* uses **k-means clustering**. K-means clustering groups the descriptors into k mutually exclusive clusters. The resulting clusters are separated by similar characteristics, where each cluster center represents a visual word.

2. The extracted features from the function are then added into the machine learning model. In MATLAB, the *Classification Learner Application* is a standard built-in application that is used to run a variety of machine learning algorithms. The *Classification Learner* performs automated data training and investigates the best suitable classification model for our Labanotation dataset. In our case, the Ensemble Bagged Tree model had the highest accuracy of 81%. This major advantage of the *Classification Learner Application* leads towards reduced searching time for finding the right classification model.

3. The Ensemble Bagged Tree classification algorithm trained the classifier on the training set. Section 4.5 describes the evaluation stage of this algorithm.

```

Input = database Labanotation symbol images
Split database: 70% training set; 30% test set

For every Labanotation image: ①
    Extract Visual Bag of Words
    Count Visual Words
Display histogram Visual Words and occurrence ②

Open Classification Learner Application
    Run all machine learning algorithms
    Find highest accuracy
Display best suitable machine
learning algorithm for training set

Train training set ③
    Use new extracted features
End

Test accuracy on test set
    Predict Outcome
Print percentage accuracy
Display Confusion Matrix

```

Figure 6. Pseudocode of Object Recognition

4.4.2 Object Recognition with Transfer Learning

The second algorithm Transfer Learning uses a deep learning approach. With Transfer Learning, we select a pretrained network and use it as a starting point to learn a new task, which in our case would be predicting Laban symbol images. The pretrained network that is used, is called AlexNet. AlexNet has already been trained on more than a million images and is able to identify and classify images into over a thousand distinct object categories, such as pencils, keyboards, and many animals (Shelhamer, 2017). The main advantage of using the Transfer Learning algorithm, is the ability to train a classifier model using relatively few labeled data, in combination with a pretrained network called AlexNet. The pretrained AlexNet network saves time in the training cycle and also significantly reduces computer resources needed.

Figure 7 below represents the pseudocode of this algorithm which is described as follows:

1. Load AlexNet and insert the database of Labanotation images. After splitting the database into the training set and test set, the structure of the AlexNet is analyzed.
2. AlexNet is designed to classify 1000 objects, however, now we need to classify Labanotation symbol images. The first step is to replace the last three layers of the pretrained network with a set of layers that is able to classify the corresponding categories of Labanotation symbols. In order to classify four categories, we have to add one additional layer to the network.
3. Furthermore, making small adjustments in choosing the weight for the InitialLearningRate. We have to adjust the learning rates for the new layer we added, so that they change faster than the rest of the network. This way earlier layers do not change that much, and we quickly learn the weights of the newer layer.

4. Train the network on the training set. AlexNet takes a random Laban symbol image as a starting point and outputs a label for this Laban symbol image, together with the probabilities for each of the possible object categories. Eventually, test the accuracy of the trained classifier of the test set, as described in Section 4.5.

```
Load AlexNet
Input = database Labanotation symbol images
Split database in 70% training set and 30%
test set (1)

Analyze structure of AlexNet

Replace last 3 layers: (2)
Layers = net Layers(1 till end -3);
Add one additional layer

Modify Learning Rate (3)
Choose weights: Initial Learning Rate
Adjust new added layer

Train AlexNet on training set (4)
Print possibilities of each category

Test classifier on test set
Predict outcome
Print percentage accuracy
Display Confusion Matrix
```

Figure 7. Pseudocode for Transfer Learning

4.4.3 Deep Learning as a feature extraction

The third algorithm, Feature Extraction using AlexNet, extracts image features from a pretrained Neural Network, as the same procedure as the Section above. Those same features are then used to train the Laban symbol image classifier. Figure 8 represents the pseudocode of this algorithm which is described as follows:

1. After loading AlexNet, the database of Labanotation symbols is inserted. After splitting the database into the training set and test set, the structure of the AlexNet is analyzed.

2. The first layer of the network has learned to filter blob and edge features. These features are then processed by deeper network layers, which combine the early features to form higher level image features. These higher-level features are better suited for recognition tasks because they combine all the primitive features into a richer image representation.

3. By using the *activations* method, you can extract features from one of the deeper layers. The layer we want to extract is named 'fc7', and we use the training features of that layer.

4. The function *fitcecoc* is an *error-correcting output codes* (ECOC) classifier for multiclass learning, where the classifier consists of multiple binary learners such as support vector machines (SVMs). The function *fitcecoc* stores the training data and the parameter values. As a result of *fitcecoc*, the deep learning network picks a random image as an input and outputs a label for the object in the image, together with the probabilities for each of the object categories. Finally, test the trained classifier with the test set, using the trained SVM model, as further described in Section 4.5.

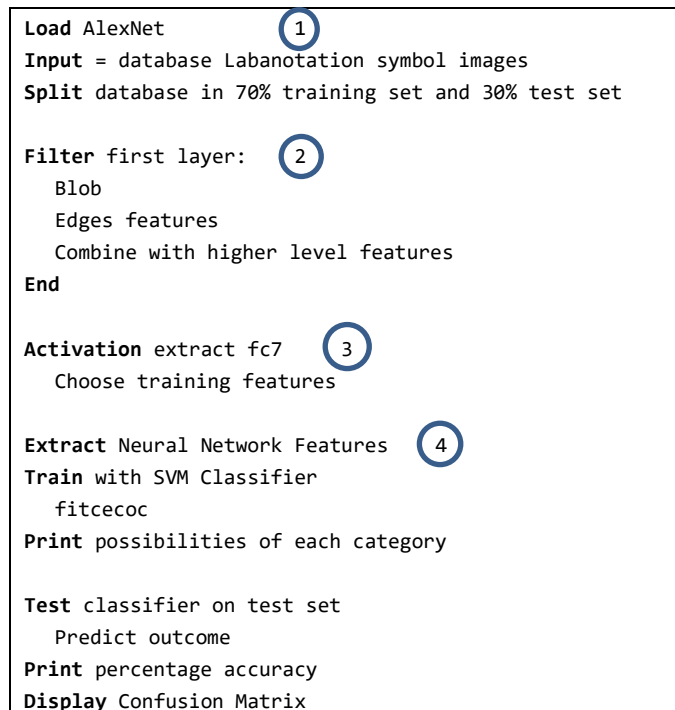


Figure 8. Pseudocode for Deep Learning as feature extraction

4.5 Evaluation stage

Now we evaluate the results of the three algorithms of Part I that were applied to the test set of the Laban database images. We performed a k-fold cross-validation. K refers to the number of groups that a given data sample is to be split into (Brownlee, 2018). According to James et al. (2017), setting $k = 10$ is commonly used in k-fold cross-validation, since the lower k is, the more biased. On the contrary, the higher k, the

higher the variability. Thus, each algorithm has been executed ten times and the results are represented in the Table 2 below.

Table 2. Accuracy in percentage obtained from the algorithms to identify Laban symbols

	<i>Object Recognition with Machine Learning</i>	<i>Transfer Learning using AlexNet</i>	<i>Feature Extraction using AlexNet</i>
<i>1e run</i>	62%	67%	77%
<i>2e run</i>	25%	72%	79%
<i>3e run</i>	20%	63%	72%
<i>4e run</i>	32%	61%	83%
<i>5e run</i>	30%	74%	82%
<i>6e run</i>	27%	69%	84%
<i>7e run</i>	30%	72%	75%
<i>8e run</i>	33%	54%	79%
<i>9e run</i>	29%	67%	78%
<i>10e run</i>	18%	72%	75%
<i>Average %</i>	30.6%	67.1%	78.4%

We calculated the average of each algorithm. The feature extraction using AlexNet algorithm has the highest accuracy with a value of 78.4%.

Each machine learning algorithm is more in-depth evaluated by using two important evaluation metrics: **recall** and **precision**. While recall refers to the percentage of the total relevant results that are correctly classified by our algorithm; precision refers to the percentage of our results which are actually relevant.

We calculated of each algorithm the precision and recall from the confusion matrix, based on the highest accuracy of Table 2. The results are depicted below in Table 3.

Table 3. Recall and precision of all three algorithms

<i>Category</i>	<i>#1 Algorithm</i>		<i>#2 Algorithm</i>		<i>#3 Algorithm</i>	
	Recall	Precision	Recall	Precision	Recall	Precision
<i>Direction Left</i>	0.23	0.20	1	0.94	1	1
<i>Left Forward</i>	0.45	1	0.46	0.67	0.67	0.67
<i>Place</i>	0.21	0.20	0.81	0.72	0.91	0.73
<i>Right Forward</i>	0	0	0.62	0.44	0.64	0.76

Remarkably, in the first algorithm, the recall and precision score are both zero for the class *Right Forward*. Next to that, based on the second and third algorithm, the highest recall and precision lies in the category *Direction Left*. The lowest precision and recall scores lie in the categories *Left Forward* and *Right Forward*. Interestingly, two categories *Right Forward* and *Left Forward* are almost identical, besides the fact that they are both reversed relative to each other, could influence the rest.

In the first algorithm, most errors were made in predicting the category *Place*, but the correct category was *Direction Left*. In the second algorithm, most misclassifying cases were in the category *Right Forward* for the category *Left Forward*. In the third algorithm, most errors were also made in predicting category *Right Forward* were it

supposed to be *Left Forward*. In conclusion, as shown in Table 3, indicates that the third algorithm shows the highest scores for all the four categories.

5 Results

5.1 #1 Algorithm: Object Recognition with Machine Learning

An interesting finding is found in the parameter ‘VocabularySize’, which means the number of visual words, and also in the parameter ‘StrongestFeatures’, which means the number of strongest features that can be extracted. Normally, increasing both parameters would increase the accuracy, but now both parameters have a maximum value of twenty. In the case of our dataset, Laban symbol images, it is relatively complex to identify ‘visual words’ and afterwards classifying them to their corresponding label. The lack of creating adequate enough words to distinguish between Laban symbols, has subsequently led to the lowest accuracy score.

5.2 #2 Algorithm: Object Recognition with Transfer Learning

Using Transfer Learning, you have the opportunity to produce small adjustments in the algorithm by changing the training options. The first training option is the ‘Initial-LearningRate’, which means how fast the transferred layers can learn. After trial-and-error, the best suitable setting to train the classifier is to set the initial learning rate to a small value of 0.001. The reason for doing so, is to slow down the learning process in the transferred layers, in such a way that the classifier will be trained longer. On the other hand, an increased learning rate is desired for the fully connected layers, in order to increase the learning speed in the new final layers at the final stage. Another training option is ‘AddingLayers’, it provides the ability to add or remove layers in order to classify new Laban images. The best suitable setting for this training option, is by adding two additional layers. An interesting finding worth noting is the fact that Algorithm 2 has a more than double increase in accuracy. This is attributed to the fact that the second algorithm makes use of the pretrained network, AlexNet. AlexNet has already been trained on more than a million images and is able to identify and classify images into over a thousand distinct object categories.

5.3 #3 Algorithm: Object Recognition with Feature Extraction

The third algorithm consists of Feature Extraction and also AlexNet. One of the main advantages is that AlexNet greatly reduces training time and also is rather straightforward to use. Unfortunately, the algorithm is a semi black-box solution, which makes it difficult to understand the code in detail. On top of that, makes it also harder to customize or optimize the algorithm. However, as stated in Table 1, this algorithm has the highest accuracy score out of all. One of the reasons for this, is that it is faster and requires less effort than Transfer Learning. In Transfer Learning you have to fine-tune the network before it can be used.

6 Part II: Recognizing Multiple Labanotation Symbols

In Part II of the research, we investigate a fourth algorithm that is capable of detecting multiple Labanotation symbols in a whole Labanotation Document. Part II is different than in Part I, with regard to two aspects: (1) Part II uses three evaluation files: one being a handwritten Labanotation document, and two evaluation files being computer-generated; (2) Part II does not use a machine learning algorithm. This fourth algorithm can only detect the Labanotation symbols that were implemented in the Labanotation database (Table 1) and cannot recognize other existing Labanotation symbols.

6.1 Engineering Stage

Prior to the development of the fourth algorithm, we insert the same Labanotation database image files that were used in the Research Part I in Section 4.3 (Table 1). The Labanotation database image files consists of 240 images. Figure 8 below represents the pseudocode of this algorithm which is described as follows:

1. In the engineering stage the function SURFFeatures is used to detect unique points in each image. The function SURFFeatures will also detect unique points in the original Labanotation document (evaluation file). The function StrongestFeatures makes sure it keeps in mind the strongest unique points of each image. Bay et al. (2008) stated that using SURFFeatures method was the best option to detect Object in images in MATLAB.
2. Based on the matched unique points in the Labanotation symbol images (input) and Evaluation Labanotation documents (output) the corresponding Laban symbols are being detected with a color box around it. Eventually, the algorithm prints whether is has found a match or not.

```
Input = database Labanotation symbol images

For every Labanotation symbol image ①
    Detect SURFFeatures
    Select StrongestFeatures
End

For every Evaluation Labanotation file
    Detect SURFFeatures
    Select StrongestFeatures
End

Match Points ②
    Every Laban symbol image
    Evaluation Laban file
Display matches
Draw Box around Labanotation symbol
    Print "Match"
Else:
    Print "No Match"
```

Figure 9. Pseudocode of Multiple Labanotation symbol Detector

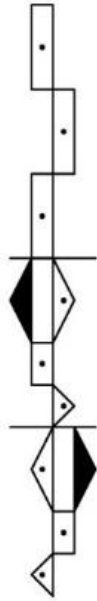
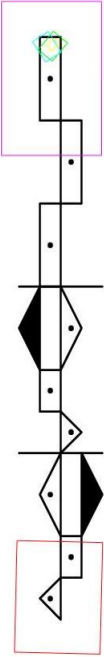
6.2 Evaluation Stage

After the engineering stage, we again continue to the evaluation stage, where we test the accuracy of the fourth algorithm, based on three Labanotation documents. The first self-created Labanotation document in LabaNotator, which is a tool that created computer-generated Labanotation files.

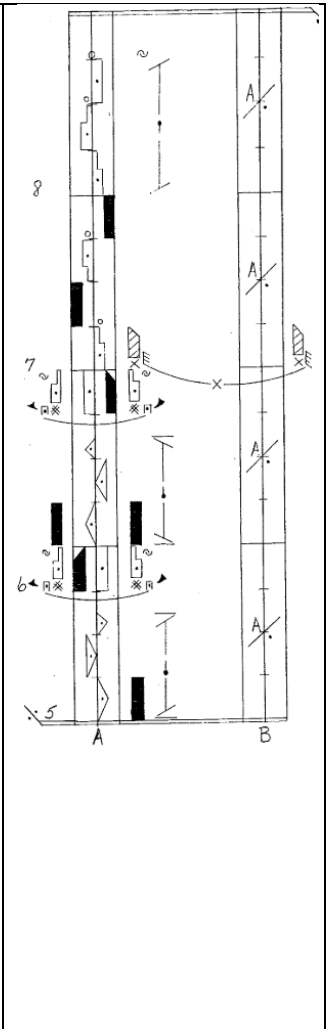
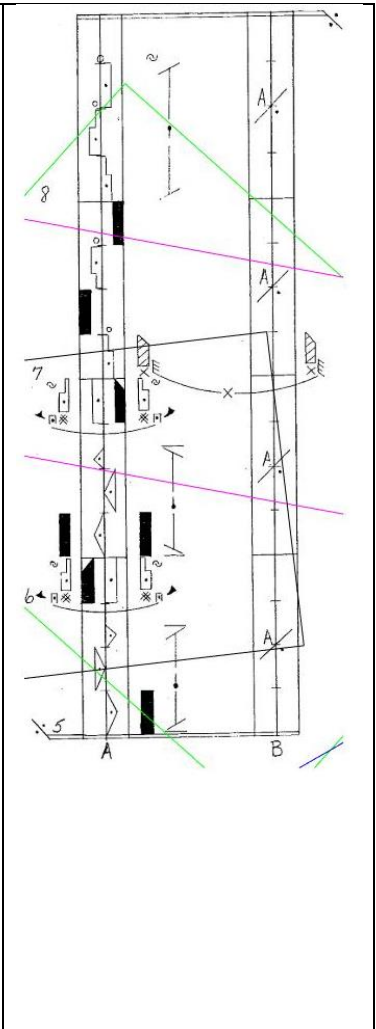
The second Labanotation document is also computer-generated file, but copied from a published paper (Wilke et al., 2005). This Labanotation file is more complex, consisting of more Laban symbols and has also unknown symbols

The third Labanotation document is a handwritten document, to test whether it also recognize something. The handwritten Labanotation document is published in the paper including the description (Gainer, 1970). For clarification, Table 4 below depicted the three evaluation files.

Table 4. Evaluation files of the fourth algorithm. First and second file are computer-generated, the third file is handwritten.

LabanID	Evaluation file	Description	Answer
L13		<p>{Place} {Place} {Place} {Direction Left} {Unknown} {Place} {Unknown} {Direction Left} {Unknown} {Place} {Direction Left}</p> <p>Total of 11 symbols, where 3 are unknown. They do not exist in the Labanotation database (Table 1). Thus, in total 8 symbols, which 2 are correct.</p> <p>50% accurate</p>	

<p>L14</p>		<p>{Left Forward} {Right Forward} {Left Forward} {Right Forward} {Left Forward} {Right Forward} {Direction Left} {Direction Left} {Left Forward} {Right Forward} {Right Forward} {Place} {Right Forward} {Place} {Left Forward} {Direction Left} {Direction Left} {Right Forward} {Unknown} {Place} {Place} {Place} {Place}</p> <p>Total of 23 symbols, where 1 is unknown. This one does not exist in the Labanotation data- base (Table 1). Thus, in total 22 sym- bols, which 1 are correct.</p> <p>4,5% accurate</p>	
------------	--	---	--

L15		<p> {Place} {Left Forward} {Right Forward} {Place} {Unknown} {Place} {Right Forward} {Place} {Left Forward} {Right Forward} {Direction Left} {Direction Left} {Direction Left} {Place} {Place} {Left Forward} {Place} {Right Forward} {Direction Right} {Direction Right} {Direction Right} {Place} </p> <p>Total of 22 symbols, where 1 is unknown. This one does not exist in the Labanotation database (Table 1). Thus, in total 21 symbols, which 0 are correct.</p> <p>0% accurate</p>	
-----	--	--	---

The accuracy is calculated based on the correctly detected Laban symbols, divided by the total number of present Laban symbols as depicted in Table 5.

Table 5. Accuracies on the multiple symbol detector algorithm

	Algorithm:	Average Accuracy (%):
<i>Self-created computer-generated in LabaNotator</i> <i>Computer-generated Original Published</i> <i>Handwritten Original Published</i>	Evaluation file #1	50 %
	Evaluation file #2	4,5 %
	Evaluation file #3	0 %

6.2 Results

As shown in Table 5 above, the highest accuracy is shown in the first evaluation file (L13). The first evaluation file is self-created, but still computer-generated in LabaNotator. As depicted in Table 4, L13 had two matches namely the category *Place* and the category *Direction Left*. Interestingly, the first match is on the first symbol of L13 and the second match is on the last symbol of L13. Both matches have the advantage of having fewer neighbors of other Labanotation symbols.

The second highest accuracy comes from the computer-generated original published file with 10%. As shown in Table 4, L14 had only one match namely the category *Left Forward*. The advantage of this match is that this Labanotation symbol stands alone and does not make any contact with other Labanotation symbols.

The lowest accuracy score is the handwritten original published file of 0%. The L15 evaluation file is negatively affected by unknown Labanotation symbols. For clarification, the unknown Labanotation symbols are depicted in Figure 10 below, these are snapshots of the L15. As result of these unknown Labanotation symbols it further limits the ability to draw a box around the actual Laban symbol. This combination of findings provides some support for the conceptual premise that avoiding unknown Labanotation symbols must be considered into a further study.

A reason why there is such a low score is the limited Labanotation database image files. Meaning, that the evaluation file consists Laban symbols that have not been seen before by our algorithm, and therefore cannot be classified. Consequently, the algorithm is only able to detect the inserted Labanotation symbols, not other created Labanotation symbols.

Interestingly, is to see how our fourth algorithm actually deals with real cases. The added value of this research will not only contribute to detecting multi-Laban symbols, but also to a broader aspect in dealing with similar symbol recognition cases, such as Chinese characters.

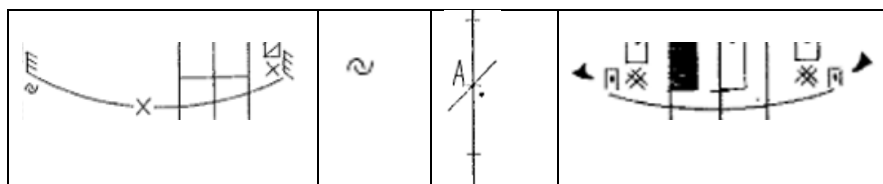


Figure 10. Unkown Labanotation files.

7 Design Features for Optical Laban Recognition (OLR) Tool

Returning to the Research Question posed at the beginning of this study, the most important design feature is creating a larger and more diverse Laban image database. Now only four categories are in there, but there are at least more than a thousand Labanotation symbols. The quality of the image database affects not only the accuracy of each algorithm, but more importantly also the precision as well as the recall. Now,

the categories *Left Forward* and *Right Forward* are too similar and therefore must be rendered with a higher pixel resolution, in order to increase image quality.

Other criteria for selecting the design features are as follows: (1) all input files must have the format of .JPG or .PNG; (2) the output provides the corresponding Laban symbol category with the respective accuracy percentage; (3) By the virtue of the research conducted in Part I, we can establish the usefulness of a pretrained neural network feature. Therefore, we could suggest the use of a more advanced pretrained neural network, namely: ResNet-50. ResNet-50 is a neural network of fifty layers deep, whereas AlexNet, which has been used in this study, has a neural network of only 12 layers deep. Another important design feature based on the results of Part II, is the implementation of a function that is able to ignore any noise or blur that are both present now in the original handwritten Labanotation files. On top of that, a function that could possibly ignore the vertical and horizontal lines, would also be desirable.

Together these results both provide insights into how an OLR Tool would look like. Unfortunately, these findings from this study cannot be extrapolated to all written symbol languages yet.

8 Discussion

8.1 Implication on theory

In our engineering study of Part I, we revealed three patterns concerning the applied machine learning algorithms: (1) the use of the feature extraction AlexNet scores the highest validation accuracy percentage; (2) the highest error is found in prediction the categories *Left Forward* and *Right Forward*; (3) the quality of the self-created image Laban dataset must be improved in order increase the accuracy. Dodge & Karam, (2016) established that the quality of the input that is used in the machine learning algorithms solely depends on the input. The outcomes of this research enrich the design features of Optical Image Recognition. Furthermore, the Optical Labanotation Recognition Tool adds value towards choreographers and other dancers that find it difficult to read and understand the Labanotation symbols.

8.2 Implication on practice

Our study findings highlight the importance of preserving the cultural heritage, mostly the dance area and making the society aware of the new possibilities with an Optical Image Recognition program. The Optical Image Recognition program is interesting because it adds value towards choreographers and other dancers that want to learn and work with the written Labanotation language. The Optical Image Recognition program is a start towards reading all Labanotation documents automatically and give an output that understand this notation. Choreographers could play a large role in this, but also the dancers themselves and in similar developing areas may use this study to formulate better dance strategies for increasing the creativity for choreographers.

8.3 Limitations

The first limitation was not having the opportunity to use a pre-labelled dataset. We concluded in our research Part I and Part II that the quality of the Laban symbol images is extremely important in identifying the Laban symbols. The second limitation was the small self-created image dataset. In further research, the training samples must be creased to a value of 1000 Laban images.

In our research we now only use the computer-generated files to recognize the Laban symbols. Since these images are computer-based and not handwritten documents, it could explain why the results of Part II have a low accuracy in the two handwritten original Labanotation files. Other factors that were not included in our study, which might have influenced our results in Part II, are the noise and blur segments. These factors are not considered at all in the fourth algorithm, and there might explain why it has such a low accuracy. An interesting direction for future research would be to consider handwritten Labanotation files and characteristics noise in the documents.

9 Conclusion

The research of Part I was conducted for designing an Optical Labanotation Recognition Tool that is able to identify individual Laban symbols. Three different machine learning algorithms were developed and evaluated based on their accuracy, recall and precision. The deep learning approach as a feature extraction using AlexNet scored the highest accuracy, with 78.4%. The results of the recall and precision evaluation metrics state that the highest score lies in the category *Direction Left*. The most errors were found in the predicted category *Left Forward*, towards the actual class *Right Forward*, facing challenges such as too many similarity issues regarding the Laban images.

The research of Part II concerns a fourth algorithm that is able to detect multiple Laban symbols in one original handwritten Labanotation document, and also in two computer-generated Labanotation documents. Both evaluation files that are computer generated are having a higher accuracy than the original handwritten Labanotation document.

10 Appendix

10.1 LabanXML

```
<laban>
  <attribute>
    <time>
      <beat>4</beat>
    </time>
  </attribute>
</notation>
<measure num="0">
```

```

        <relationship type="grasp">
            <left>
                <hand>
                    <bodypart>hand</bodypart>
                </hand>
            </left>
        </relationship>
    </measure>
</notation>

```

Figure. 7 Short piece of LabanXML code (Nakamura & Hachimura, 2006).

11 References

- Aher, S. R., & Kapale, P. N. D. (2017). Automatic Number Plate Recognition System for Vehicle Identification Using Optical Character Recognition. *International Reserach Journal of Engineering and Technology*, 04(06), 3–6. <https://doi.org/10.1109/ICETC.2009.54>
- Aventina, R. (1928). Labanotation - Wikipedia. Retrieved May 27, 2019, from <https://commons.wikimedia.org/wiki/File:Labanotation1.jpg>
- Baloh, M., Bohak, C., & Marolt, M. (2015). StreamingLaban – Low bandwidth format for dance encoding with Labanotation, (January).
- Barbacci, S. (2002). Labanotation: a universal movement notation language. *ISAS*. Retrieved from [https://jcom.sissa.it/sites/default/files/documents/jcom0101\(2002\)A01.pdf](https://jcom.sissa.it/sites/default/files/documents/jcom0101(2002)A01.pdf)
- Bay, H., Tuytelaars, T., & Gool, L. Van. (n.d.). *SURF: Speeded Up Robust Features*. Retrieved from <https://www.vision.ee.ethz.ch/~surf/eccv06.pdf>
- Bezjak, P. (n.d.). LabaNotator - Home. Retrieved January 16, 2019, from <http://www.labanotator.com/>
- Blake, J. (2000). On Defining the Cultural Heritage. *Journal of Urban Affairs*, 42(1), 17. <https://doi.org/10.3868/s050-004-015-0003-8>
- Brownlee, J. (2018). A Gentle Introduction to k-fold Cross-Validation. Retrieved May 21, 2019, from <https://machinelearningmastery.com/k-fold-cross-validation/>
- Calvert, T., Wilke, L., Ryman, R., & Fox, I. (2005). Applications of computers to dance. *IEEE Computer Graphics and Applications*, 25(2), 6–12. <https://doi.org/10.1109/MCG.2005.33>
- Chung, H.-S., Kim, J.-M., Byun, Y.-C., & Byun, S.-Y. (2005). Retrieving and Exploring Ontology-Based Human Motion Sequences. *Springer- Verlag Berlin Heidelberg*, 3482, 788–797.
- Csurka, G., Dance, C. R., Fan, L., Willamowski, J., & Bray, C. (2004). *Visual Categorization with Bags of Keypoints*. Retrieved from <https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/csurka-eccv-04.pdf>
- Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *IEEE Computer Society*, 1, 886–893. <https://doi.org/10.1109/CVPR.2005.177>
- de Boer, V., Jansen, J., Tjon-A-Pauw, A. L., & Nack, F. (2018). Interactive dance choreography assistance. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10714 LNCS, 637–652. https://doi.org/10.1007/978-3-319-76270-8_45
- Dodge, S., & Karam, L. (2016). *Understanding How Image Quality Affects Deep Neural Networks*. Retrieved from <https://arxiv.org/pdf/1604.04004.pdf>
- Dyke, J. Van. (2001). Intention: Questions regarding its role in choreography. *Journal of Dance Education*, 1(3), 96–101. <https://doi.org/10.1080/15290824.2001.10387186>
- El Raheb, K., & Ioannidis, Y. (2012). A labanotation based ontology for representing dance movement. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7206 LNAI, 106–117. https://doi.org/10.1007/978-3-642-34182-3_10
- Fügedi, J. (2006). *Basics of Laban Kinetography for Traditional Dancers*. Budapest: Institute for Musicology, Research Centre for the Humanities.
- Gainer, J. (1970). Folk Collection No 2.pdf. Dance Notaiton Bureau.
- Hansen, J. (2002). A Matlab Project in Optical Character Recognition (OCR). *DSP Lab*.
- Hariharan, A. (2018). How to Use Machine Learning to Predict the Quality of Wines. Retrieved February 22, 2019, from <https://medium.freecodecamp.org/using-machine-learning-to-predict-the-quality-of-wines-9e2e13d7480d>

- Hat, J. (2006). Movementxml : a Representation of Semantics of Human Movement Based on Labanotation. *Channels*.
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition*. Retrieved from <http://image-net.org/challenges/LSVRC/2015/>
- Huster. (2007). Labanotation parties du corps. Retrieved May 27, 2019, from <https://commons.wikimedia.org/wiki/File:Labanotation1.JPG>
- ImageNet. (n.d.). [Http://www.image-net.org](http://www.image-net.org).
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). *An Introduction to Statistical Learning: with Applications in R (Springer Texts in Statistics)*. Retrieved from <https://machinelearningmastery.com/k-fold-cross-validation/>
- Johnson, C. J., & Snyder, A. F. (1999). Securing Our Dance Heritage: Issues in the Documentation and Preservation of Dance. *Council on Library and Information Resources Washington, D.C.* Retrieved from <https://files.eric.ed.gov/fulltext/ED437893.pdf>
- King, K. (2011). How many songs are there in the world? 97 million and counting. | MarsBands. Retrieved May 20, 2019, from <http://www.marsbands.com/2011/10/97-million-and-counting/>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*. Retrieved from <http://code.google.com/p/cuda-convnet/>
- Maheswari, J. P. (2018). Breaking the curse of small datasets in Machine Learning: Part 1. Retrieved February 22, 2019, from <https://towardsdatascience.com/breaking-the-curse-of-small-datasets-in-machine-learning-part-1-36f28b0c044d>
- MathWorks, I. T. (2019a). Pretrained Neural Networks. Retrieved from <https://nl.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html?jsessionid=e287acabfdbc52e26035b745f67>
- MathWorks, I. T. (2019b). What is MATLAB? - MATLAB. Retrieved January 16, 2019, from <https://nl.mathworks.com/discovery/what-is-matlab.html>
- Nakamura, M. (2006). AN XML REPRESENTATION OF LABANOTATION, LabanXML, AND ITS IMPLEMENTATION ON THE NOTATION EDITOR LabanEditor2, 9(May), 47–51.
- Nakamura, M., & Hachimura, K. (2006). AN XML REPRESENTATION OF LABANOTATION , LabanXML , AND ITS IMPLEMENTATION ON THE NOTATION EDITOR LabanEditor2, 9, 47–51.
- Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). *Reading Digits in Natural Images with Unsupervised Feature Learning*. Retrieved from <http://ufldl.stanford.edu/housenumbers/>
- Noonan, D., Mountney, P., Elson, D., Darzi, A., & Yang, G.-Z. (2009). A Stereoscopic Fibroscope for Camera Motion and 3D Depth Recovery During Minimally Invasive Surgery. *Www.Sciweavers.Org*, pp. 4463–4468.
- Rebelo, A., Capela, G., Cardoso, J. S., Capela, G., & Cardoso, J. S. (2010). Optical recognition of music symbols A comparative study, 13, 19–31. <https://doi.org/10.1007/s10032-009-0100-1>
- Rebelo, Ana, Fujinaga, I., Paszkiewicz, F., Marcal, A. R. S., Guedes, C., & Cardoso, J. S. (2012). Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3), 173–190. <https://doi.org/10.1007/s13735-012-0004-6>
- Rio, F. Del, Messina, P., Dominguez, V., & Parra, D. (2018). *Do Better ImageNet Models Transfer Better... for Image Recommendation?* Retrieved from www.ugallery.com
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. Retrieved from <http://image-net.org/challenges/LSVRC/>
- Sankhla, A., Kalangutkar, V., Bhuyan, H. B. G. S., Mallick, T., Nautiyal, V., & B, P. P. Das. (2018). Automated Translation of Human Postures from Kinect Data to Labanotation (Vol. 841, pp. 494–505). Springer Singapore. <https://doi.org/10.1007/978-981-13-0020-2>
- Shelhamer, E. (2017). BVLC AlexNet Model. Retrieved from https://github.com/BVLC/caffe/tree/master/models/bvlc_alexnet
- Szegedy, C., Liu, W., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., ... Rabinovich, A. (2014). *Going deeper with convolutions*. Retrieved from <https://arxiv.org/pdf/1409.4842.pdf>
- Tiwari, S., Mishra, S., Bhatia, P., & Yadav, K. (2013). Optical Character Recognition using MATLAB. *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, 2(5), 2278–2909. Retrieved from <http://ijarece.org/wp-content/uploads/2013/08/IJARECE-VOL-2-ISSUE-5-579-582.pdf>
- Tongpaeng, Y., Sureephong, P., Rattanakhum, M., & Yu, H. (2017). *Thai dance knowledge archive framework based on Labanotation represented in 3D animation. 2nd Joint International Conference on Digital Arts, Media and Technology 2017: Digital Economy for Sustainable Growth, ICDAMT*

2017. <https://doi.org/10.1109/ICDAMT.2017.7904936>
- Tsirliganis, N., Pavlidis, G., Koutsoudis, A., Papadopoulou, D., Tsompanopoulos, A., Stavroglou, K., ... Chamzas, C. (2004). Archiving Cultural Objects in the 21st Century. *Journal of Cultural Heritage*. <https://doi.org/10.1016/j.culher.2004.04.001>
- Vilbrandt, C., Pasko, G., Pasko, A., Fayolle, P.-A., Vilbrandt, T., Goodwin, J. R., ... Kunii, T. L. (2004). Cultural Heritage Preservation Using Constructive Shape Modeling. *COMPUTER GRAPHICS Forum*, 23(1), 25–41. Retrieved from https://s3.amazonaws.com/academia.edu.documents/31792773/VPP04_HeritConst_CGF.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1544988284&Signature=YFerr4VA9BG9c0aLy2NG5Pd0IZ8%3D&response-content-disposition=inline%3Bfilename%3DCultural_Heritage_Preserv
- Wang, K., Babenko, B., & Belongie, S. (2011). End-to-end scene text recognition. *Proceedings of the IEEE International Conference on Computer Vision*, (November 2011), 1457–1464. <https://doi.org/10.1109/ICCV.2011.6126402>
- Wikipedia contributors. (2018). Performing arts. Retrieved from https://en.wikipedia.org/wiki/Performing_arts
- Wilke, L., Calvert, T., Ryman, R., & Fox, I. (2005). From dance notation to human animation: The LabanDancer project. *Computer Animation and Virtual Worlds*, 16(3–4), 201–211. <https://doi.org/10.1002/cav.90>