Delaunay Triangulation Data Augmentation guided by Visual Analytics for Deep Learning

Alan Z. Peixinho*, Bárbara C. Benato*, Luis G. Nonato† and Alexandre X. Falcão*
*Institute of Computing, University of Campinas, Campinas, Brazil
Email: alan-peixinho@hotmail.com, barbarabenato@gmail.com and afalcao@ic.unicamp.br
†Institute of Mathematical and Computer Sciences, University of São Paulo, São Carlos, Brazil
Email: gnonato@icmc.usp.br

***Abstract*—It is well known that image classification problems can be effectively solved by Convolutional Neural Networks (CNNs). However, the number of supervised training examples from all categories must be high enough to avoid model overfitting. In this case, two key alternatives are usually presented (a) the generation of artificial examples, known as data augmentation, and (b) reusing a CNN previously trained over a large supervised training set from another image classification problem — a strategy known as transfer learning. Deep learning approaches have rarely exploited the superior ability of humans for cognitive tasks during the machine learning loop. We advocate that the expert intervention through visual analytics can improve machine learning. In this work, we demonstrate this claim by proposing a data augmentation framework based on Encoder-Decoder Neural Networks (EDNNs) and visual analytics for the design of more effective CNN-based image classifiers. An EDNN is initially trained such that its encoder extracts a feature vector from each training image. These samples are projected from the encoder feature space on to a 2D coordinate space. The expert includes points to the projection space and the feature vectors of the new samples are obtained on the original feature space by interpolation. The decoder generates artificial images from the feature vectors of the new samples and the augmented training set is used to improve the CNN-based classifier. We evaluate methods for the proposed framework and demonstrate its advantages using data from a real problem as case study — the diagnosis of helminth eggs in humans. We also show that transfer learning and data augmentation by affine transformations can further improve the results.**

I. Introduction

Given an image training set with examples from all categories, we wish to train a Convolutional Neural Network (CNN) for image classification [1]–[3]. We know that CNNs with deep neural architectures can be very successful in image classification, as well as in other applications. However, they often require a high number of supervised samples per category to avoid *model overfitting* — i.e., high classification accuracy on the training set with low accuracy on unseen test sets. Regularization techniques, such as *dropout* methods, are usually applied to amend the problem [4], [5]. In addition to dropout, two important alternatives are (a) *data augmentation* (the generation of artificial examples) [6]–[8] and (b) transfer learning (the weight refinement of a CNN previously trained over a large supervised training set from another image classification problem) [9]–[13].

In this work, we present a framework for data augmentation that exploits two directions:

1) the capability of Encoder-Decoder Neural Networks (EDNNs, also known as auto-encoders) to extract image features and reconstruct an approximation of the same image with no label supervision [14]–[17];
2) the superior ability of humans in understanding, by a suitable projection, the distribution of the samples in a given feature space to augment the number of supervised examples.

Our approach is inspired on recent works that have exploited visual analytics to train pattern classifiers [18], understand deep learning [19], and improve deep neural networks [20].

From a small training set with images from all categories, we first train an EDNN in order to extract image features for sample projection (Figure 1). The training samples are projected from the encoder feature space onto a 2D coordinate space using some suitable dimensionality reduction algorithm [21]–[24]. The expert includes points to the projection space and the feature vectors of the new samples are obtained in the original feature space by interpolation, where we proposed a triangulation based interpolation method on the encoded feature space. The decoder generates artificial images from the feature vectors of the new samples and this augmented training set is used for the design of a CNN-based image classifier. Therefore, by adding points to each category on the projection space, the user is indirectly creating examples of images, as synthesized by the decoder, with slight texture and color variations with respect to the original ones. We advocate that this data augmentation process guided by visual

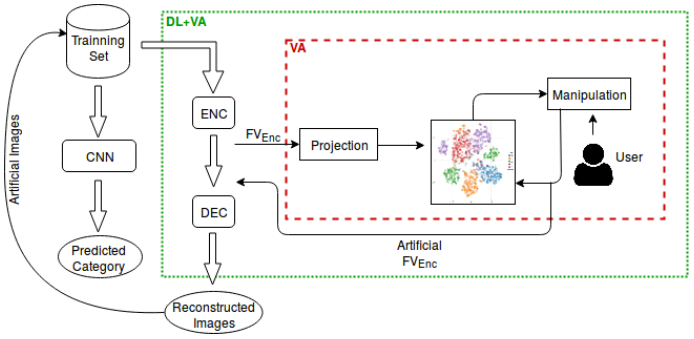

Fig. 1. An overview of the data augmentation system guided by visual analytics for deep learning.

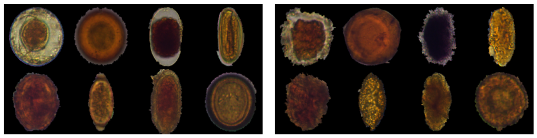

Fig. 2. Real images (left) of helminth eggs, one for each specie, and examples of similar impurities (right).

analytics can improve the design of the CNN-based classifier.

In addition, transfer learning [10]–[12] can considerably improve the effectiveness of the CNN training. It requires the adaptation of the input and output layers to the problem, which might not be possible always. Data augmentation by affine transformations are among the most commonly used techniques [25], [26], but they can also impair the performance of the CNN-based classifier if wrongly chosen [13]. We evaluate the impact of transfer learning and the most suitable affine transformation to further improve our results. The advantages of our framework are demonstrated by using data from a real problem as case study — the diagnosis of helminth eggs in humans. In this application, we count on an automated system that reads and segments objects from images of optical microscopy slides, separates those objects into three groups of intestinal parasites (helminth eggs and similar impurities; protozoa cists and similar impurities; and helmith larvae and similar impurities), extracts image features, and classifies the objects of each group according to their specie or as fecal impurity [27], [28]. We use examples of the eight most common species of helminth eggs in Brazil and their similar fecal impurities (Figure 2). Impurities that differ from all species of parasites are previously eliminated by image processing. Even so, the number of similar impurities in a microscopy slide can be from tens to hundreds times higher than the number of parasites. In our training set, for instance, the less prevalent species really require data augmentation. The objects must also be aligned by their principal axis at the center of the image, which helps the CNN to cope with variations in object position.

II. Related works and relevance of this contribution

The combined use of visual analytics and machine learning techniques has mostly been focused on assisting the user to understand, extract information, and/or take actions in a given application. Jeong et al. [29] developed a visualization system to understand Principal Component Analysis. The system allows the user to select a cluster of data items in one coordinate space and visualize the corresponding items highlighted. Choo el al. [30] presented a visual analytics system that uses Linear Discriminant Analysis for feature space reduction and allows the user to visualize data clusters and decide the class of new examples. Heidemann [31] presented an inter-active learning system that combines the user's mental model of the class definition with a classification model trained by examples to facilitate the detection and correction of label inconsistencies.

In neural networks, visual analytics has played a role in understanding how deep learning creates effective classification systems. Rauber et al. [19] studied CNNs by visualizing their neuron activity during machine learning. They observed, for instance, the increasing separation among categories on dynamic t-SNE projections [24] of the hidden layers of a CNN as they approach the last layer. Rauber et al. [18] also proposed a visual analytics system to select the most relevant features for the design of pattern classifiers. Liu et al. [32] presented a visual analytics system to help machine learning experts in the understanding, diagnosis, and refinement of CNN-based classifiers. Cashman et al. [33] purposed an interactive tool to visualize the gradient flow during the training of Recurrent Neural Networks. Pezzoti et al. [20] presented a progressive visual analytics system that supports the design of deep neural networks (DNNs) by showing the link between filters and patterns they detect on training.

In regard to the deep learning models called generative, due to their independence of supervised samples, there are specific networks: Deep Boltzmann Machines [34], Generative Adversarial Networks [35], and Encoder-Decoder Neural Networks [14]–[17]. We have chosen EDNNs, since the encoder can reduce an input image into a feature vector for a suitable projection while the decoder can reconstruct an artificial image, similar to the original ones, from an interpolated feature vector.

Despite the number of data augmentation techniques [25], [26], [36], we could not find methods that use the decoder of an EDNN and not even visual analytics for data augmentation. There are methods that create artificial data to train a more effective EDNN [36], but not methods that decode feature vectors artifically created. Therefore, as far as we know, the present work is seminal in the generation of artificial feature vectors by the user, as guided by visual analytics, and in the conversion of those features into an artificial image.

III. Data Augmentation guided by Visual Analytics

Figure 1 has been used to shortly introduce the proposed framework for data augmentation guided by visual analytics. In this section, we provide more details about each operation.

A. Convolutional Neural Network (CNN)

The framework relies on a CNN to extract image features and classify patterns. A CNN usually consists of convolutional neural layers (which include neural activation and might include other non-linear operations, such as pooling and normalization), fully-connected neural layers, and a decision layer [3]. The role of the convolutional layers is to extract a high dimensional and sparse feature vector from each input image. We call this the *convolutional feature vector* (FV_{CNV}), which will be useful in one of the experiments for data augmentation. Fully-connected layers can play the role of feature space reduction with neuron specialization [19] and, finally, the decision layer outputs the category of the input image.

In principle, one could use any type of effective neural network for image classification. We decided for CNNs, such as AlexNet [2], GoogleNet [37], VGG-16 [38], ResNet [39], and DenseNet [40], due to their success in a wide variety of applications [41]–[43].

In order to reduce the chances of overfitting our model, the number of supervised examples per category (images with their true label) must be high. In many scientific applications, however, this is not always possible, due to high acquisition costs, high annotation costs, or even the scarcity of available data. This actually defines two types of problems for our framework: (a) small supervised training sets, which is the one addressed in this work, and (b) large and mostly unsupervised training sets with a low number of supervised samples per category. In (a), data augmentation is crucial. In (b), one may benefit of a label propagation procedure from supervised to unsupervised samples. In our framework, the user could act for label propagation guided by visual analytics. However, even in (b), might exist categories with a few samples asking for data augmentation. Therefore, for the sake of simplicity, we will focus on (a) only.

CNNs can be trained by *filter learning*, *architecture learning*, or *transfer leaning* [13]. In architecture learning, the network weights are randomly selected, under certain constraints, and some algorithm finds the best hyperparameters (e.g., number of layers, sizes of the filters in each layer, stride in pooling, etc.) [44]. In filter learning, the weights are randomly initialized and subsequently optimized by backpropagation. Transfer learning, on the other hand, uses as weight initialization the resulting network trained for another image classification problem with plenty of supervised samples per category, where a fast weight refinement by backpropagation usually suffices to train the network for a new image classification problem. However, this approach, might requires the images to be cropped and/or resized to adapt the data to the input of the network, also the output layer must be adapted to the required number of categories. Architecture learning is not largely used as filter and transfer learning, but it can usually provide better results. However, filter learning and transfer learning is consistently the best approach to cope with scenarios with small data [13]. We assess the performance of our framework using the two most common approaches, *filter learning* and *transfer learning*. In any case, some dropout method [4], [5] is part of the training protocol of a given CNN, working as regularizer to reduce the chances of overfitting. Roughly, these methods simply set to zero the weights of connections or the output values of neuron activations randomly selected during training, thus avoiding the network to strongly rely in a small set of neurons.

B. Projections from nD to 2D

By exploiting suitable projections from nD to kD, where $n > k, k = 2, 3$, one can analyze and interact with the projected samples in order to understand the structure of their distribution in nD. We found simpler and yet effective to work with 2D and nonlinear projections by using algorithms such as t-SNE [21] and LAMP [22]. An advantage in using LAMP is that users can rearrange the position certain points in the visual space in order to steer the projection. As we will see later, the ability to interact with the layout has implications to the subsequent data augmentation step (Section III-D).

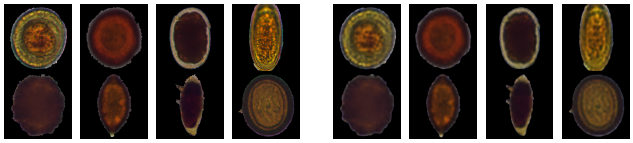

Fig. 3. Eight examples of original images (left) and their reconstruction (right) by a convolutional EDNN.

Once data is projected in the visual space, new samples can be drawn in 2D based on the position of the projected data. The new sampled points are mapped back to the feature space from which images are synthesized so as to build the augmented data. An EDNN is required to encode real image feature vectors for projection and decode artificial image feature vectors from user manipulation.

C. Encoder-Decoder Neural Network (EDNN)

An EDNN is an unsupervised machine learning approach that relies on a criterion function to encode an input image into a compressed feature vector and decode the feature vector into an approximated reconstruction of that image. It can rely on fully-connected and/or convolutional neural layers for this task. We define the *encoder feature vector* (FV_{ENC}) for sample projection as the one produced at the output layer of the encoder (Figure 1).

Figure 3 shows examples of input and output images of an EDNN. We have evaluated several types of EDNNs in regard to their capability to (a) create feature spaces where the categories are separated from each other as best as possible (in nD and consequently in 2D, as shown in [18]) and (b) decode images with slight variations of color and texture (not critical distortions). Several types have succeeded according to [14], [17], but the convolutional EDNN [16] was the best according to both criteria. We believe this is related to the small size of the training set, which makes easier to estimate weights for considerably less neuron connections. Since the encoder feature vectors are not used to help image classification (which is an interesting issue to be addressed in a future work), we decided for convolutional EDNNs. Figure 3 shows examples of artificial images decoded for data augmentation by this type of EDNN. Although some detail is lost, the reconstructed images are still highly resembling its original input.

D. Manipulation

The main idea in manipulation is to add new samples, especially to the smaller categories. By data augmentation, we expect a considerably higher number of artificial samples after manipulation. The expert must then identify confident regions to insert 2D points from a given category. By doing that, the corresponding nD feature vectors of the artificial samples must be interpolated by taking into account the nD

feature vectors of the original training samples. Manipulation can also involve rearrangement of the samples in 2D for data augmentation, such that the user can further separate the categories. For instance, the LAMP technique [22] allows users to drag some points (control points) so as to bring closer the points belonging to the same class, steering the projection to reflect the user provided arrangement.

1) Delauney Triangulation-based Data Augmentation: The Radial Basis Function interpolation is often applied in multi-dimensional data interpolation [45]. However, the interpolated data has led in our application to saturation in the features, thus saturating the decoder image reconstruction as well. An Example of saturated image can be found in Figure 4.

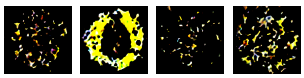

Fig. 4. Examples of saturated images obtained after decoder reconstruction of RBF interpolated samples.

Therefore we followed a different approach for data interpolation that is better suited for our problem. Devising an automatic method to detect confident regions, add points in 2D, as well as interpolate them back to the high dimensional nD space.

Let $X = \{x_1, \ldots, x_m\}$ be a set of nD points and $Y = \{y_1, \ldots, y_m\}$ be the image of X in 2D obtained via multidimensional projection. The two-dimensional layout can be interactively tweaked by users so as to better group the classes in the visual space. Given X and Y, the proposed triangulation-based data augmentation methodology operates in three steps: triangulation, new data generation, and data interpolation.

The first step, *triangulation*, builds the Delaunay triangulation from the points in Y and labels triangles according to the vertices' class. Specifically, a triangle is labeled as belonging to a class 'A' if all its three vertices are labeled as 'A'. Otherwise the triangle is labeled as invalid. The left image in Figure 5 illustrates labeled (colored) and invalid triangles for a given input Y.

The second step draws a set of s points from bivariate Gaussian distributions centered in each point from Y with covariance given by σI, where I is the 2×2 identity matrix and σ is the length of the diagonal of the bounding box of Y. Sample points lying inside a labeled triangle is considered for the augmentation process while points inside invalid triangles or outside the triangulation hull are disregarded. Points considered for the augmentation are labeled according to the triangle they lie in. Figure 5 right depicts points considered for the augmentation when $s = 20$ points per Gaussian are drawn.

Let p_y be a point considered for augmentation and $t_y = [y_r, y_s, y_t]$, $y_r, y_s, y_t \in Y$ the triangle containing p_y. The triangle t_y has a counterpart triangle $t_x = [x_r, x_s, x_t]$ $x_r, x_s, x_t \in X$ in the original nD space, where y_r, y_s, y_t are the image of x_r, x_s, x_t in the visual space. Therefore, p_y can be mapped back to the nD space by simply computing a point $p_x \in t_x$ with the same barycentric coordinates as $p_y \in t_y$. The third step of the proposed augmentation technique repeats the process above for all points considered for augmentation, giving rise to a new set of points in the nD space that is input in the decoder to generate the sought augmented data.

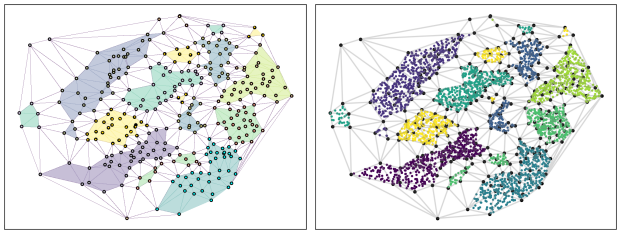

Fig. 5. Delauney triangulation of training data (left), and augmented samples inside valid triangles (right).

TABLE I
NOMENCLATURE ADOPTED TO DESCRIBE EXPERIMENTS.

S0	Helminth eggs **without impurities**.
S1	Helminth eggs **with impurities** (the real problem).
T0	CNN trained by **filter learning**.
T1	CNN trained by **transfer learning**.
A0	Original image training set (i.e., set without data augmentation).
A1	Augmented training set by visual analytics using **t-SNE** and **triangulation-based interpolation**.
A2	Augmented training set by visual analytics using **LAMP** and **triangulation-based interpolation**.
A3	Augmented training set by visual analytics using **LAMP**, **sample rearrangement**, and **triangulation-based interpolation**.
A4	The same as **A3**, but **without impurity projection**.
A5	The same as **A3**, but **concatenating** FV_{ENC} **and** FV_{CNV} for sample projection.
A6	Augmented training set by **random translations from A0**.
A7	Augmented training set by **random translations from A5**.

IV. Experimental Setup

This section describes the dataset of helminth eggs, the architectures of the EDNN and CNN, and the data augmentation methods used to evaluate our framework. The nomenclature adopted in the following sections are described in Table I.

A. Image Dataset of Helminth Eggs

All experiments are performed in a dataset containing $12,691$ images of helminth eggs, already segmented, aligned, and centered at the image. This dataset contains an unbalanced numbers of samples from eight species of eggs and similar fecal impurities, as the ninth category (Table II).

The number of impurities similar to each of those species is very high, which makes the problem atypical. The impurities appear as *white noise* all over the projection of the other categories. Some species, such as *H.diminuta* and *E.vermicularis*, are also low in number. This makes data augmentation essential. However, typical image classification problems are better represented by the case we exclude impurities from the dataset. Therefore, we define two scenarios to evaluate the framework

TABLE II
IMAGE DATASET OF HELMITH EGGS.

Species	# samples
H.nana	501
H.diminuta	83
Ancilostomideo	286
E.vermicularis	103
A.lumbricoides	835
T.trichiura	435
S.mansoni	254
Taenia	379
Impurities	9,535

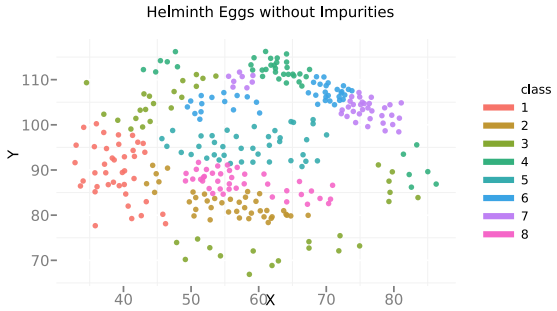

Fig. 6. LAMP 2D projection of dataset without impurities. User was allowed to rearrange control points to better group the classes.

S0 A typical scenario in general image classification problems, as defined by the dataset of helminth eggs with no impurities (Figure 6).

S1 The real scenario of our application, as defined by the dataset of helminth eggs with impurities (Figure 7).

In both scenarios, in order to simulate small supervised training sets, we randomly selected 40 training images per category, totalizing 320 samples for S0 and 360 samples for S1. For network training, it is also important that the training sets are balanced. The remaining samples were used to test the accuracy of the system on unseen samples (the unbalanced test set).

B. The Architecture of the EDNN

Some experiments were guided to select the number of convolution layers and the number of filters in each convolution

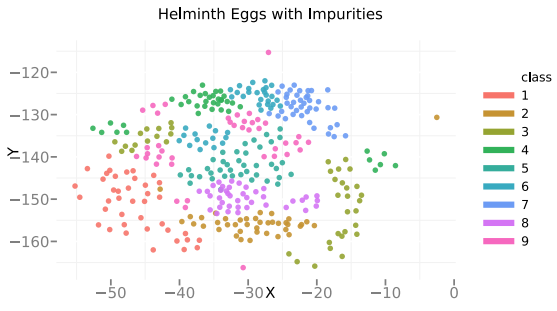

Fig. 7. LAMP 2D projection of dataset with impurities. The impurities class is divided in many clusters in the whole space. User was allowed to rearrange control points to better group the classes.

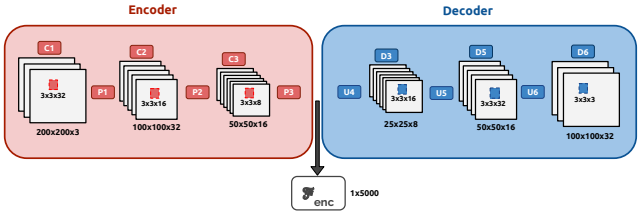

Fig. 8. Convolutional EDNN used for the experiments.

layer. First, we compared the created feature space with the categories separated from each other and the decoded images, while we were adding one more layer. Considering just one layer for encode, the separability of feature space and the decoded images not performed well. Thus, we considered 2 and then 3 layers and the separability seems adequate. A higher number of layers could require a pre-trainning method as initialization. To choose the number of filters we began with 8 filters per layer and we observed the decoded images. We increased the number of filters per layer and the reconstructed images presented color and texture more similar with the original images. In this way, we have considered 32, 16 and 8 filters per layer respectively.

The architecture of the convolutional EDNN used in all experiments is given in Figure 8. We denote Convolutional (C), Max Pooling (P), Upsampling (U) and Fully Connected layers (FC), all interleaved by Relu activation units, except by the last layer that presented Sigmoid activation units. The Sigmoid activation unit at the last layer is responsible for generating different values in range $[0, 1]$ and result in the output image.

An ordinary normalization was made with the images before the training, i.e., we made a normalization that could recover a RGB image later. Thus, we divided each pixel of image by 255. We trained the convolutional EDNN during 1000 epochs and refined the network weights by backpropagation. We used the Mean Squared Error as cost function. We also tested the Cross Entropy as cost function, but the resulting images seemed saturated. In order to obtain the reconstructed images by the synthetic feature vector, we saved the network weights.

C. The Architecture of the CNN

Given its simplicity, we decided to develop our CNN based on AlexNet, whose architecture is presented in Figure 9. We denote Convolutional (C), Normalization (N), Max Pooling (P) and Fully-Connected layers (FC), all interleaved by Relu activation units.

This network has been trained by filter and transfer learning, depending on the compared method. In filter learning, we initialized the CNN with an xavier distribution, as described in [46], as it is well known to help convergence on Deep Networks. For transfer learning, we used the weights provided in the Caffe framework [47] for initialization. In both cases 160 epochs were used to train/refine weights by backpropaga-

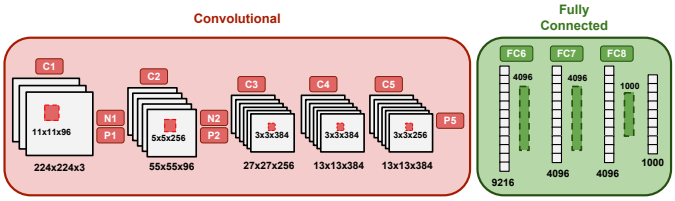

Fig. 9. The architecture of the AlexNet [2].

tion, with a starting learning rate of 0.001 (0.0001 for transfer learning) reduced by a factor of 0.1 each 40 epochs.

D. Compared Methods

We have evaluated our framework for the training sets of eggs without (S0) and with (S1) impurities, trained the CNN by filter (T0) and transfer (T1) learning, and used each data augmentation method below from A0 to A7. Any data augmentation approach should be able to obtain higher accuracy on the test set in comparison with A0, which represents the original training set S0 or S1 without data augmentation. We observed, for instance, that data augmentation via rotation and scaling degrade the performance of the CNN, making the results worse than A0 for both S0 and S1. This should be expected since we have mentioned the importance of aligning the objects by their principal axis at the center of the image to help the CNN with variations in object position. However, the nuclei of the cells can appear anywhere inside them. This makes important to slightly and randomly translate the objects from the center of the image for more effective data augmentation. In this case we translate the images from a uniform random distribution of $[-20, 20]$ pixels in both dimensions. Methods A6 and A7 illustrate that, being A7 based on visual analytics. Methods A1 and A2 compare different projection algorithms. The comparison between A2 and A3 shows the importance of the interpolation technique. A3 indicates the importance of rearranging samples on the projection space to better separate the categories.

In order to facilitate reasoning, we have tried a few tricks to improve visualization in the scenario S1. Method A4 does not project impurities (Figure 6), so the user adds only artificial samples of parasites. In method A5, the convolutional feature vector (FV_{CNV}) of the CNN trained by transfer learning is concatenated to the encoder feature vector (FV_{ENC}) to improve visualization with impurity projection (Figure 10). After interpolation, FV_{ENC} is recovered for the generation of the corresponding artificial image by the decoder. Methods A6 and A7 are only tested on the real and most difficult scenario with impurities.

Our automated method for generating new samples, can potentially create different number of samples per classes. For this reason, in all cases, the augmented training sets are reduced to present the same number of samples per class for all methods. In A7, however, we evaluate further improvements of A5 when random object translations are added to the augmented training set. Evaluating the gain of combining both data augmentation strategies.

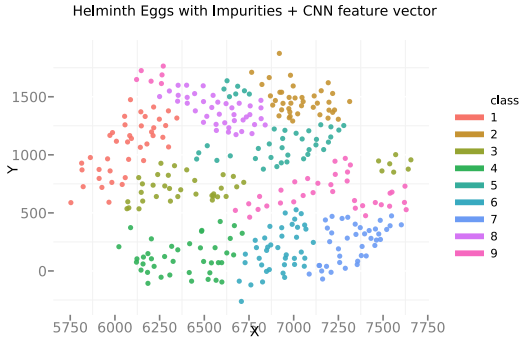

Fig. 10. LAMP 2D projection of Eggs data set concatenated with CNN features, improving the visual class separation even in the scenario with impurities, reducing the user effort to better group the classes.

TABLE III
CNN TRAINED BY **FILTER LEARNING** (T0) ON THE TRAINING SET **WITHOUT IMPURITIES** (S0).

	Kappa	Accuracy
A0	0.7734	0.8095
A2	0.9117	0.9276
A3	0.9209	0.9354

V. Experiments and Results

In this section, we present the experiments and results with the methods A0 to A7. First, we evaluate the impact of data augmentation by visual analytics and the importance of the projection and interpolation algorithms on the simpler scenario S0. Second, we evaluate our tweaks to improve visualization in the real scenario S1. Finally, we evaluate the further improvements for the real scenario S1, when combining multiple techniques to address overfitting.

A. Importance of Data Augmentation by Visual Analytics

Table III shows a toy example, where we ignore the presence of impurities, in order to assess the effectiveness of our proposed approach according to two metrics, kappa and accuracy. Where our approach shows a considerable increase for both. Comparing the effect user driven sample rearrangement, notice that the rearrangement shows a slightly better result when compared with the augmentation withouth user intervention, which indicates the advantage of using a projection algorithm that allows sample rearrangement such as LAMP.

In our experiments we are considering 1198 artificial samples added to S0 (with a new training roughly $4.5\times$ bigger). And, for S1, we have added 1598 artificial samples (training roughly $5.5\times$ bigger). The only exception lies in A7, which combines two data augmentation approaches. Figure 11 present some examples of some artificial images.

Including our augmented samples, we create a denser version of our data manifold, creating a smoother gradient in the backpropagation. Figure 12 compares the learning curve for

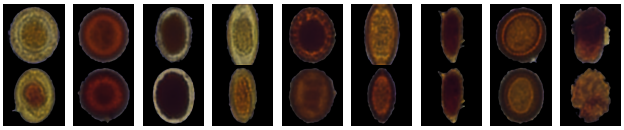

Fig. 11. Artificial images generated for experiments A4 (top) and A5 (bottom).

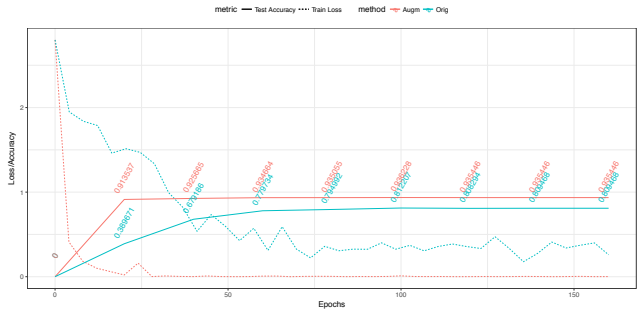

Fig. 12. The training loss and test accuracy along the epochs for both augmented and no augmented datasets.

A3 and A0. Note that, our proposed augmentation devises a faster network convergence in a smoother loss curve.

B. Impact in the Presence of a Diverse Class

In order to assess the performance of our approach, we consider here, a more challenging scenario, since the presence of impurities in the training set can really impair the effectiveness of the CNN and considerably reduce the effectiveness gain of A3 over A0 (Table IV). Reinforcing the importance of tricks to improve visualization, we evaluate a different visualization technique (A1), as well as an improvement on the features themselves. Method A4 shows that data augmentation by visual analytics can still produce considerable effectiveness gains over A0, when we focus on improving the feature separability. Finally, in A5, the combination of visual based and affine data augmentation indicates they provide uncorrelated samples, indicating its combination can improve even further the performance.

C. Further Improving Data Augmentation by Visual Analytics

Transfer learning is not always possible and data augmentation by affine transformations can, in some cases, degrade the CNN's performance. However, for this particular application, transfer learning and data augmentation by random object translations can provide considerable effectiveness gains.

TABLE IV
CNN TRAINED BY **FILTER LEARNING** (T0) ON THE TRAINING SET **WITH IMPURITIES** (S1).

	Kappa	Accuracy
A0	0.1717	0.2546
A1	0.1948	0.2831
A3	0.2872	0.4554
A4	0.2947	0.4631
A5	0.4553	0.6781

TABLE V
CNN TRAINED BY **TRANSFER LEARNING** (T1) ON THE TRAINING SET **WITH IMPURITIES** (S1).

	Kappa	Accuracy
A0	0.5432	0.7522
A5	0.5964	0.7924
A6	0.5927	0.7882
A7	0.6556	0.8329

Compare, for instance, A0 in Table V with A0 in Table IV and the gain of A6 over A0 in Table V. Indeed, for this application, A6 can produce comparable results with data augmentation by visual analytics. Note that, due to transfer learning, the differences between A5, and A6 are not relevant. However, A7 shows considerable effectiveness gain over the others when transfer learning, data augmentation by random translations, and data augmentation by visual analytics are combined.

VI. CONCLUSION

We have presented a visual analytics system to improve the performance of CNN-based image classification by data augmentation. Our system relies on a neural encoder to extract image features, projection, addition, and interpolation algorithms to create artificial supervised samples, and a neural decoder to convert them into artificial training images. We have demonstrated the advantages of the proposed system for simple and complex scenarios, as created from a real problem — the diagnosis of helminth eggs in humans. We have also demonstrated the possibility of combining our approach with other techniques that reduce overfitting.

For future work, we intend to address problems with large and mostly unsupervised training sets. We believe that the larger amount of unsupervised samples in the EDNN will improve the separation among categories on the projections. Additionaly, data augmentation will benefit with label propagation from supervised to unsupervised samples, as guided by visual analytics. The visual analytics loop can also include changes in the EDNN and CNN, which will complete the proposed framework to address image classification problems.

ACKNOWLEDGMENT

The authors are grateful to FAPESP grants #2014/12236-1 and #2016/25776-0 and CNPq grant 302643/2013-3 and 302970/2014-2. The views expressed are those of the authors and do not reflect the official policy or position of the São Paulo Research Foundation.

REFERENCES

[1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[4] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 1058–1066.

[5] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, January 2014.

[6] P. Y. Simard, D. Steinkraus, and J. Platt, "Best practices for convolutional neural networks applied to visual document analysis." Institute of Electrical and Electronics Engineers, Inc., August 2003.

[7] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "High-performance neural networks for visual object classification," *CoRR*, vol. abs/1102.0183, 2011. [Online]. Available: http://arxiv.org/abs/1102.0183

[8] D. C. Ciresan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," *CoRR*, vol. abs/1202.2745, 2012. [Online]. Available: http://arxiv.org/abs/1202.2745

[9] Y. Xia, "Fine-tuning for image style recognition," 2015.

[10] R. Caruana, "Learning many related tasks at the same time with backpropagation," in *Proceedings of the 7th International Conference on Neural Information Processing Systems*, ser. NIPS'94. Cambridge, MA, USA: MIT Press, 1994, pp. 657–664.

[11] L. Torrey and J. Shavlik., *Transfer learning*, 2009.

[12] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1717–1724.

[13] A. Z. Peixinho, "Learning image features by convolutional networks under supervised data constraint," Master's thesis, University of Campinas (UNICAMP), Brazil, Aug 2017.

[14] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, December 2010.

[15] D. Kingma and M. Welling, "Auto-encoding variational bayes," 2013.

[16] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Artificial Neural Networks and Machine Learning (ICANN 2011: 21st International Conference on Artificial Neural Networks, Proc. Part I)*. Berlin, Heidelberg: Springer, 2011, pp. 52–59.

[17] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, no. C, pp. 232–242, April 2016.

[18] P. Rauber, A. Falcão, and A. Telea, "Projections as visual aids for classification system design," *Information Visualization*, 2017.

[19] P. E. Rauber, S. G. Fadel, A. X. Falcão, and A. C. Telea, "Visualizing the hidden activity of artificial neural networks," *IEEE Transactions on Visualization and Computer Graphics (Proceedings of the Visual Analytics Science and Technology 2016)*, vol. 23, no. 01, January 2017.

[20] N. Pezzotti, T. Hllt, J. V. Gemert, B. P. F. Lelieveldt, E. Eisemann, and A. Vilanova, "Deepeyes: Progressive visual analytics for designing deep neural networks," *IEEE Transactions on Visualization and Computer Graphics (Proceedings of the Visual Analytics Science and Technology 2017)*, vol. 24, no. 1, pp. 98–108, Jan 2018.

[21] L. V. D. Maaten and G. Hinton, "Visualizing high-dimensional data using t-sne," *J. Mach. Learn. Res.*, vol. 9, no. 1, pp. 2579–2605, 2008.

[22] P. Joia, D. Coimbra, J. A. Cuminato, F. V. Paulovich, and L. G. Nonato, "Local affine multidimensional projection," *IEEE Trans. Vis. Comp. Graph.*, vol. 17, no. 12, pp. 2563–2571, Dec 2011.

[23] L. V. D. Maaten, "Accelerating t-SNE using tree-based algorithms," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3221–3245, 2014.

[24] P. Rauber, A. Falcão, and A. Telea, "Visualizing time-dependent data using dynamic t-sne," in *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers*, ser. EuroVis '16, 2016, pp. 73–77.

[25] S. Wong, A. Gatt, V. Stamatescu, and M. McDonnell, "Understanding data augmentation for classification: when to warp?" *CoRR*, vol. abs/1609.08764, 2016. [Online]. Available: http://arxiv.org/abs/1609.08764

[26] L. Taylor and G. Nitschke, "Improving deep learning using generic data augmentation," *CoRR*, vol. abs/1708.06020, 2017. [Online]. Available: http://arxiv.org/abs/1708.06020

[27] C. Suzuki, J. Gomes, A. Falcão, S. Shimizu, and J. Papa, "Automated diagnosis of human intestinal parasites using optical microscopy images," in *2013 IEEE 10th International Symposium on Biomedical Imaging*, April 2013, pp. 460–463.

[28] C. T. N. Suzuki, J. F. Gomes, A. X. Falcão, J. P. Papa, and S. Hoshino-Shimizu, "Automatic segmentation and classification of human intestinal parasites from microscopy images," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 3, pp. 803–812, March 2013.

[29] D. H. Jeong, C. Ziemkiewicz, B. Fisher, W. Ribarsky, and R. Chang, "ipca: An interactive system for pca-based visual analytics," in *Proceedings of the 11th Eurographics / IEEE - VGTC Conference on Visualization*, ser. EuroVis'09. Chichester, UK: The Eurographs Association & John Wiley & Sons, Ltd., 2009, pp. 767–774.

[30] J. Choo, H. Lee, J. Kihm, and H. Park, "ivisclassifier: An interactive visual analytics system for classification based on supervised dimension reduction," in *2010 IEEE Symposium on Visual Analytics Science and Technology*, Oct 2010, pp. 27–34.

[31] G. Heidemann, D. Weiskopf, M. Hoferlin, R. Netzel, and B. Hoferlin, "Inter-active learning of ad-hoc classifiers for video visual analytics," *2012 IEEE Conference on Visual Analytics Science and Technology (VAST 2012)*, vol. 00, pp. 23–32, 2012.

[32] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *IEEE Trans. Vis. Comp. Graph.*, vol. 23, no. 1, pp. 91–100, Jan. 2017.

[33] D. Cashman, G. Patterson, A. Mosca, and R. Chang, "Rnnbow: Visualizing learning via backpropagation gradients in recurrent neural networks," in *VADL2017: Workshop on Visual Analytics for Deep Learning at IEEE VIS 17*, Phoenix, Arizona USA, Jul 2017.

[34] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, D. van Dyk and M. Welling, Eds., vol. 5. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 16–18 Apr 2009, pp. 448–455.

[35] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 2672–2680.

[36] X. Zhang, Y. Fu, A. Zang, L. Sigal, and G. Agam, "Learning classifiers from synthetic data using a multichannel autoencoder," *CoRR*, vol. abs/1503.03163, 2015.

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015, pp. 1–9.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[40] G. Huang, Z. Liu, K. Q. Weinberger, and L. Van der Maaten, "Densely connected convolutional networks," *arXiv:1608.06993*, 2016.

[41] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, 2016.

[42] D. Wang, A. Khosla, R. Gargeya, H. Irshad, and A. H. Beck, "Deep learning for identifying metastatic breast cancer," *arXiv preprint arXiv:1606.05718*, 2016.

[43] S. A. Bargal, E. Barsoum, C. C. Ferrer, and C. Zhang, "Emotion recognition in the wild from videos using images," in *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. ACM, 2016, pp. 433–436.

[44] D. Cox and N. Pinto, "Beyond simple features: A large-scale feature search approach to unconstrained face recognition," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*. IEEE, 2011, pp. 8–15.

[45] G. B. Wright, "Radial basis function interpolation: numerical and analytical developments," 2003.

[46] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[47] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, ser. MM '14. New York, NY, USA: ACM, 2014, pp. 675–678.