Deep learning approach for classifying, detecting and predicting photometric redshifts of quasars in the Sloan Digital Sky Survey stripe 82

Johanna Itam-Pasquet, Jérôme Pasquet

A&A 611, A97 (2018)
https://doi.org/10.1051/0004-6361/201731106

Astronomy & Astrophysics

Deep learning approach for classifying, detecting and predicting photometric redshifts of quasars in the Sloan Digital Sky Survey stripe 82⋆

J. Pasquet-Itam[1,2] and J. Pasquet[3,4]

[1] LUPM UMR 5299 CNRS/UM, Université de Montpellier, CC 72, 34095 Montpellier Cedex 05, France
[2] CPPM, CNRS-IN2P3, Université Aix Marseille II, CC 907, 13288 Marseille Cedex 9, France
e-mail: pasquet@cppm.in2p3.fr
[3] LIRMM UMR 5506 - team ICAR, Université de Montpellier, Campus St Priest, 34090 Montpellier, France
[4] LSIS UMR 7296, CNRS, ENSAM, Université de Toulon et Aix-Marseille, Bâtiment Polytech, 13397 Marseille, France
e-mail: jerome.pasquet@lsis.org

Received 4 May 2017 / Accepted 3 November 2017

ABSTRACT

We have applied a convolutional neural network (CNN) to classify and detect quasars in the Sloan Digital Sky Survey Stripe 82 and also to predict the photometric redshifts of quasars. The network takes the variability of objects into account by converting light curves into images. The width of the images, noted w, corresponds to the five magnitudes *ugriz* and the height of the images, noted h, represents the date of the observation. The CNN provides good results since its precision is 0.988 for a recall of 0.90, compared to a precision of 0.985 for the same recall with a random forest classifier. Moreover 175 new quasar candidates are found with the CNN considering a fixed recall of 0.97. The combination of probabilities given by the CNN and the random forest makes good performance even better with a precision of 0.99 for a recall of 0.90. For the redshift predictions, the CNN presents excellent results which are higher than those obtained with a feature extraction step and different classifiers (a K-nearest-neighbors, a support vector machine, a random forest and a Gaussian process classifier). Indeed, the accuracy of the CNN within $|\Delta z| < 0.1$ can reach 78.09%, within $|\Delta z| < 0.2$ reaches 86.15%, within $|\Delta z| < 0.3$ reaches 91.2% and the value of root mean square (rms) is 0.359. The performance of the KNN decreases for the three $|\Delta z|$ regions, since within the accuracy of $|\Delta z| < 0.1$, $|\Delta z| < 0.2$, and $|\Delta z| < 0.3$ is 73.72%, 82.46%, and 90.09% respectively, and the value of rms amounts to 0.395. So the CNN successfully reduces the dispersion and the catastrophic redshifts of quasars. This new method is very promising for the future of big databases such as the Large Synoptic Survey Telescope.

Key words. methods: data analysis – techniques: photometric – techniques: image processing – quasars: general – surveys

1. Introduction

Quasars are powered by accretion onto supermassive black holes at the dynamical centers of their host galaxies, producing high luminosities spanning a broad range of frequencies. They are of paramount importance in astronomy. For example, their studies can inform on massive blackhole (e.g., Portinari et al. 2012). Moreover, as they are the most luminous active galactic nuclei (AGN), they can be seen far across the Universe. So they give clues to the evolution and structure of galaxies (e.g., Hopkins et al. 2006). They are also used as background objects to study the absorption of intergalactic matter in the line of sight, which have many applications in cosmology (e.g., Lopez et al. 2008). With the advent of large and dedicated surveys such as the Sloan Digital Sky Survey (SDSS; York et al. 2000) and the 2dF Quasar Redshift Survey (2QZ; Croom et al. 2009), the number of known quasars has rapidly increased. Thus, the SDSS DR7 Quasar catalog (Schneider et al. 2010) contains 105 783 spectroscopically confirmed quasars. The catalog covers an area of ≃9380 deg^2 and the quasar redshifts range from 0.065 to 5.46.

With the soon coming of the Large Synoptic Survey Telescope (LSST Science Collaboration 2009), it is important to develop classification tools for quasar detection given the huge amount of future data. In this way, machine learning algorithms are being used increasingly. These algorithms permit us to predict the label of an object thanks to the extraction of different features which characterize the object (e.g., the color of the source). Several classifiers are now commonly used in astronomy like random forests which are a set of decision trees (Quinlan 1986), Naives Bayes (Duda & Hart 1973), neural networks (Rumelhart et al. 1986) and support vector machines (Cortes & Vapnik 1995). These methods are very powerful in classification and detection of variable objects in astronomy (e.g., Eyer & Blake 2005; Dubath et al. 2011; Blomme et al. 2011; Rimoldini et al. 2012; Peng et al. 2012; Peters et al. 2015). We can also cite the recent work of Hernitschek et al. (2016) on the classification and the detection of QSOs in the Pan-STARR S1 (PS1) 3π survey. This is a multi-epoch survey that covered three quarters of the sky at typically 35 epochs between 2010 and the beginning of 2014 with five filters (g_{P1}, r_{P1}, i_{P1}, z_{P1}, y_{P1}). They use a random forest classifier and colors and a structure function as features, to identify 1 000 000 QSO candidates.

The main motivation for this work is to propose a new classification and a detection method for quasars in the Sloan Digital Sky Survey Stripe 82, that can be easily adapted to large

⋆ A table of the candidates is only available at the CDS via anonymous ftp to cdsarc.u-strasbg.fr (130.79.128.5) or via http://cdsarc.u-strasbg.fr/viz-bin/qcat?J/A+A/611/A97

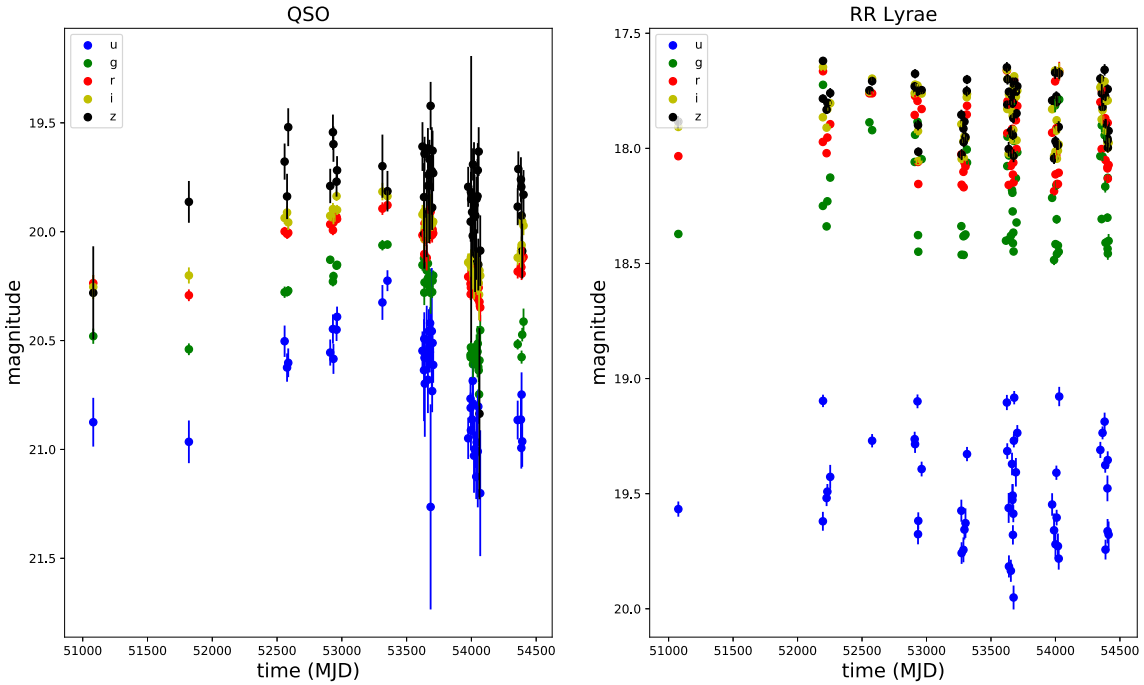

Fig. 1. Two examples of variable object types in the UWVSC catalog. In the *left panel* is a quasar light curve and in the *right panel* a RR Lyrae light curve.

future surveys like LSST or DES (The Dark Energy Survey Collaboration 2005). The algorithms mentioned above, involves a feature extraction step but the set of features can be incomplete to characterize the variability of quasars. That is why we proposed to use another branch of machine learning namely deep learning. It is a supervised learning which takes raw data into account and extracts by itself the best features for a given problem. This method gives very good results in many fields. In particular we used a convolutional neural network (CNN) architecture which gives excellent results in several signal processing challenges including Imagenet (Russakovsky et al. 2015), LifeClef (Joly et al. 2016) amongst others. This approach is very recent in Astronomy and its first applications show good results, for example for the classification of galaxy morphologies from astronomical images (Huertas-Company et al. 2015).

In this work, we propose an innovative architecture based on a CNN to detect and classify quasars from light curves, thus taking into account the variability of objects. We also apply this kind of architecture to estimate the photometric redshifts of quasars. The estimation of photometric redshifts by a CNN classifier is an original method which is very promising. This paper is organized as follows. In Sect. 2, we introduce the Stripe 82 data set. In Sect. 3, we describe the CNN architecture and processing. In Sect. 4, we propose our CNN architecture for the detection and the classification of quasars. In Sect. 5, we analyze and discuss the new quasar candidates detected by our method. Then, we compare our algorithm with a random forest classifier and combine them. In Sect. 6 we propose to use a similar CNN architecture to predict the photometric redshifts of quasars. Finally we summarize our results in Sect. 7.

2. Data

The Sloan Digital Sky Survey (SDSS) is a multi-filter imaging and spectroscopic redshift survey using a dedicated 2.5-meter telescope at Apache Point observatory in New Mexico. It provides deep photometry ($r < 22.5$) in five passbands (*ugriz*). The SDSS has imaged a 2.5 degree wide stripe along the Celestial Equator in the Southern Galactic Cap several times, called Stripe 82. It is a deeper survey of 275 deg^2. It was previously imaged about once to three times a year from 2000 to 2005 (SDSS-I), then with an increased cadence of 10–20 times a year from 2005 to 2008 (SDSS-II) as part of the SDSS-II supernovae survey (Frieman et al. 2008). There are on average 53 epochs, over a time span of 5 to 10 years (Abazajian et al. 2009).

The imaging data used in our work consists of objects solely from the publicly available variable source catalog (UWVSC; Ivezić et al. 2007; Sesar et al. 2007) constructed by researchers at the University of Washington. This catalog contains 67 507 unresolved, variable candidates with $g \leq 20.5$ mag, at least 10 observations in both g and r bands, and a light curve with a root-mean-square scatter >0.05 mag and χ^2 per degree of freedom greater than three in both the g and r bands. Among the data, some variable objects have been identified, they are essentially quasars and pulsating stars (see Fig. 1). However a large part of the data consists of unknown variable objects. In this way this catalog is interesting to identify new variable objects. This catalog and all light curves are publicly available.[1] We use the UWVSC as the basis for learning and testing for several reasons: 1) it contains over 9000 known spectroscopically confirmed quasars (Meusinger et al. 2011) whose the distribution of redshifts is shown in Fig. 2 ; 2) it is a robust variable catalog with a good photometry; 3) the catalog is a useful testbed for time domain science to prepare future data sets like LSST.

[1] http://www.astro.washington.edu/users/ivezic/sdss/catalogs/S82variables.html

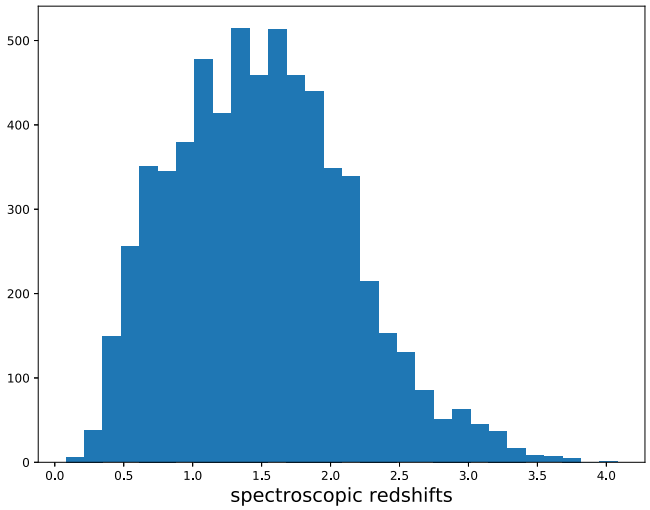

Fig. 2. Distribution of spectroscopic redshifts of known quasars in the UWVSC catalog.

3. The convolutional neural networks

In this work, we are particularly interested in CNN, which are an approach to deep learning methods and are proving their worth in many research fields.

3.1. Light curve images

As a CNN takes images as input, we had to find a way to convert a light curve to an image taking into account the variability of objects which gives a crucial information for the classification of variable objects. Thus, we propose to create images whose the width is represented by the five magnitudes (u, g, r, i, and z), and the height corresponds to the date of the observation. In Stripe 82, there are a maximum of 3340 days of observation so images should have a dimension of 5×3340 pixels. However processing these images is very costly in VRAM memory, so we divided the time interval of a light curve by averaging the observations taken on two consecutive days, so as to get images of dimensions of 5×1670 pixels. Then 60 pixels were appended to the edges of the image to avoid side-effects. Therefore the size of the final images, called hereafter, LCI (Light Curve Images), is 5×1700 pixels.

In order to increase the robustness of the network, the learning needs to be free of the positions of points. To do this, we generated new light curves by making time translations for all points on a given light curve. Thus only the global shape of the light curve is taken into account and no positions of points are considered as more important than others. This process is similar to that of classical data augmentation (Le Guennec et al. 2016; Krizhevsky et al. 2012) in the CNN learning and increases the size of the database by a factor of 13.

3.2. Introduction to the CNN

An artificial neuron is a computational model inspired by natural neurons. A natural neuron is an electrically excitable cell that processes and transmits information via synapses which are connected with other cells. When the signal received is strong enough (higher than a specific threshold), the neuron is activated and emits a signal which might activate other neurons. Artificial neurons do not reproduce the complexity of real neurons, but the global structure is quite similar. Indeed, input data are multiplied by weights and then computed by a mathematical function which determines the activation of the neuron. An artificial neural network is then composed of different neuron layers connected with each other. A layer of rank n takes as input the output of the layer of rank $n-1$. The nomenclature of layers is the following:

i) a layer whose input is not connected to the output of another layer, but to the raw data is called input layer;
ii) a layer whose output is not connected is called an output layer;
iii) a layer is called hidden if it has either input or output. Each layer is composed of several tens of thousands of neurons. In the specific case of a CNN, neurons perform convolution (see Sect. 3.2.1) and pooling operations (see Sect. 3.2.2). Layers are sequentially processed as follows: first, a convolution operation is applied on raw input data; then the output signal is modified by a nonlinear function; finally a pooling operation can be processed. Note that the output of a layer could be considered as a set of images. In CNN terminology, each image has named feature map. After all convolution and pooling layers, the last convolution layer is connected to a succession of layers called fully connected layers and operating as a classical neural network. The last one uses a softmax operation to give a probability that the input light curve is either a quasar light curve or another object. To perform the learning phase, parameters of convolution and fully connected layers are tuned using a stochastic gradient descent specific to the given problem, in this case the recognition of quasar light curves. This optimization process is very costly, but it can be highly parallelizable. We use the Caffe (Jia et al. 2014) framework to train our CNN. The results are obtained using a GTX Titan X card, packed in 3072 cores with a 1 GHz base.

3.2.1. Convolution

If we consider a layer or a set of feature maps as input, the first step is to apply convolutions. For the first layer, the convolution is done between the input image and a filter. Each filter leads to a filtered image. Convolution layers are composed of convolutional neurons. Each convolutional neuron applies the sum of 2D convolutions between the input feature maps and its kernel. In the simple case where only one feature map is passed into the input convolutional neuron, the 2D convolution between the K kernel of size $w \times h$ and the input feature map $I \in \mathbb{R}^2$ is noted $I * K$ and is defined as: $(I * K)_{x,y} = \sum_{x'=x-\frac{w}{2}}^{x+\frac{w}{2}} \sum_{y'=y-\frac{h}{2}}^{y+\frac{h}{2}} K_{x'+\frac{w}{2}-x,\, y'+\frac{h}{2}-y} I_{x',y'}$ with (x, y) the coordinates of a given pixel into the output feature map.

In the case of convolutional neural networks, a neuron takes as input each of p feature maps of the previously layer noted I^l with $l \in \{0...p\}$. The resulting feature map is the sum of p 2D convolutions between the kernel K^l and the map I^l (see Fig. 3) and is defined as

$$(I * K) = \sum_{l=0}^{p} (K^l * I^l). \quad (1)$$

In this work we propose to use two types of convolutions that we call temporal convolutions and filter convolutions. The temporal convolutions use a kernel with a x-dimension of 1 pixel, so the five magnitudes (u, g, r, i, and z) are convoluted separately, and with a y-dimension variable in the interval {5, 11, 21, 41} pixels. Thus the temporal convolutions take into account a value of magnitude at different times and at different resolutions.

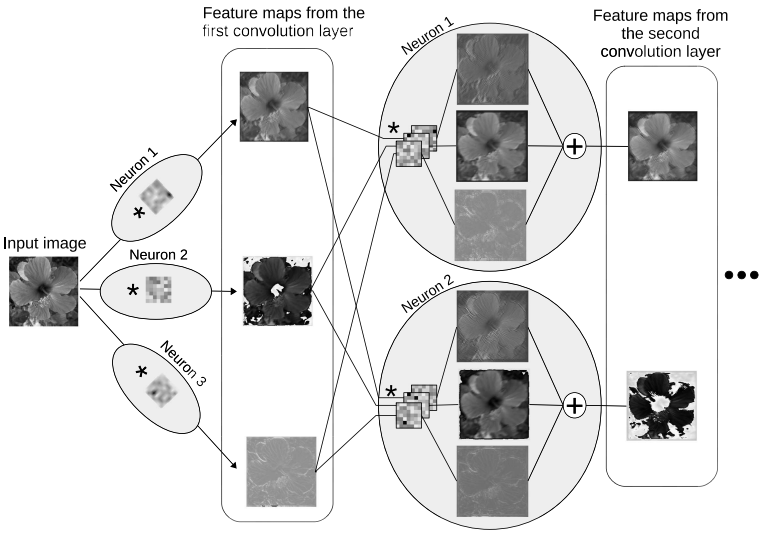

Fig. 3. Representation of two convolution layers of a network. The first layer is composed of 3 neurons making a convolution between the input image and their kernels. The second layer includes two neurons making a sum of convolutions as defined in Eq. (1).

The advantage of this type of convolutions is to create a network which is able to detect short and long variability patterns. The filter convolutions use a kernel with a dimension of 5×1 pixels, so they merge the values of the five magnitudes in order to integrate the information from color which is an important feature to characterize variable objects, at a given time.

3.2.2. Pooling

The network can be composed of pooling layers which quantify the information while reducing the data volume. The pooling layer operates independently on each feature map. On each feature map, it slides a specific filter which represents the local distribution. The two most used methods consist in selecting only the maximal or the mean value of the data in the local region. As the observational data are not continuous in time, several pixels in the LCI are equal to zero. Thus we decide to adapt the pooling by the mean, that is, we do not take into account the null pixels in the computation of the mean. Our architecture includes this improvement of the pooling on the first pooling layers of the network. The others layers contain a max pooling.

3.2.3. Activation functions

The convolution layers are followed by nonlinear transformations whose goal is to solve the nonlinear classification problems. The two most used functions are the Rectify Lineair Unit (ReLU; Nair & Hinton 2010) defined by $f(x) = \max(x, 0)$ and the hyperbolic tangent. In our network, to saturate the input signal we apply a hyperbolic tangent function on all of the first convolution layers. The other layers use a PReLU (He et al. 2015) function defined as: $f(x) = \begin{cases} \alpha x & x < 0 \\ x & x \geq 0 \end{cases}$, with α an hyperparameter defined by back-propagation.

4. Our CNN architecture

The overall structure can be subdivided into successive processing blocks at different temporal resolutions (4, 8, 16, 32, and 64 days) as shown in Fig. 4, on five levels of depth. The initial resolution of LCI is two days per pixel. At each processing level, this resolution is reduced by a factor of two, by a max-pooling. Moreover, each processing block is powered by a set of feature maps coming from an average-pooling retrieved a parallel on the first light curve image. These feature maps are then convoluted by three types of temporal convolutions, processing images by three different filter sizes. This set of temporal convolutions is similar to a multi-resolutions process which is used in modern architectures like the GoogleNet network (Szegedy et al. 2015). The resulting feature maps are then transmitted to the processing block of the associated resolution. We note $\mathbb{F}_i$ the set of resulting feature maps transmitted to the processing block i. Thus as shown in Fig. 4, feature maps in which one pixel represents four days are transmitted to module A, and those in which one pixel represents eight days are transmitted to module B, and so on.

In a first step, a filter convolution (MC1 in schema 4) is applied on $\mathbb{F}_A$. The feature maps resulting from MC1 are noted $\mathbb{F}'_A$. We then apply two temporal convolutions (TC7 and TC8) with shared weights. This term can be explained by the difference of the convolution kernels applied to the sets $\mathbb{F}_A$ and $\mathbb{F}'_A$. We modify the standard convolutions so a given kernel can be applied on two sets of feature maps with different sizes. The goal is to highlight similar temporal patterns between two sets of feature maps $\mathbb{F}_A$ and $\mathbb{F}'_A$, and so between mixed and not-mixed magnitudes amongst themselves. The feature maps coming from the convolution layers TC7 and TC8 on the set $\mathbb{F}_A$ are concatenated and then pooled in the pooling layer P4. The resulting feature maps are noted Ω_B and then transmitted to the processing block B. The same process is applied to the set of feature maps $\mathbb{F}'_A$ giving the set Ω'_B.

The second processing block performs a temporal convolution with two kernels of different sizes (TC12 and TC13) on $\mathbb{F}_B$. The resulting feature maps are then transmitted to two layers: C7 and MC2. In the C7 layer, they are concatenated with the set Ω_B. In the MC2 layer, they are convoluted by a filter. The resulting feature maps are concatenated with that of the set Ω'_B. The set of produced feature maps are temporally convoluted with shared weights in the TC14 layer. The result is then pooled and transmitted to the processing block C.

The functioning of the rest of the blocks are similar to that of block B. The number of feature maps and the size of each layer are noted in Table A.1 in the Appendix. We note that the processing block E transmits only feature maps whose magnitudes are merged, which are then temporally convoluted and transmitted to the fully connected layers. After convolution layers, the size of feature maps are 53×1 pixel.

In the network architecture we use two processes to avoid over fitting. First, all the feature maps are normalized using the batch normalization (Ioffe & Szegedy 2015). Second, the

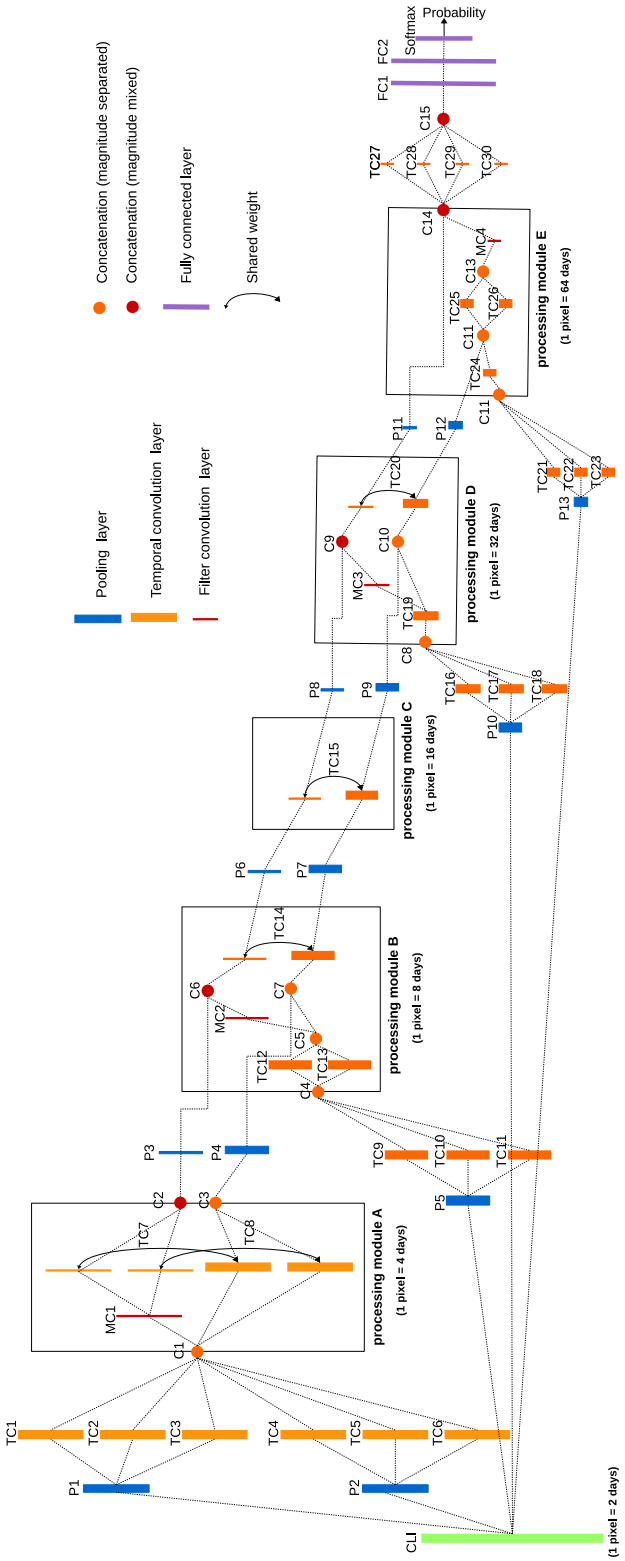

Fig. 4. Representation of the architecture that we are proposing. The structure is subdivided into five successive processing blocks at different temporal resolutions. Two types of convolutions are used: temporal convolutions with four kernel sizes: 41×1, 21×1, 11×1, and 5×1 and filter convolutions with a kernel size of 5×1.

outputs of the fully connected layers are randomly dropout (Srivastava et al. 2014). During the back-propagation processing, the network has to determine a large number of parameters, namely 1 802 032 in the convolution layers and 11 468 80 in the fully-connected layers.

5. Classification results

5.1. Experimental protocol

We did five cross-validations of the database by always selecting 75% of the LCI for the learning base and 25% for the testing base. For each of the five cross-validations, each CNN completed its learning on 60 epochs (during an epoch, each LCI is transmitted to the network and its error is back-propagated). Each CNN has three outputs on the softmax layer corresponding to the following classes: quasars, pulsating stars (RR Lyrae and Δ Scuti) and other objects. During the testing phase, each CNN gives a list of detected quasars in the testing base. We merge the lists given by each CNN into one list that we evaluate.

5.2. Results

The performance of the CNN is given in Fig. 5 in function of the magnitude and the redshift. We can notice that for a g-band magnitude below 17 magnitudes, the value of the recall decreases until a recall of 50%. It is due to the too low number of examples of very bright quasars in the training set. Indeed there are only 22 light curves of quasars in the training database with magnitudes between 15 and 17. However the recall is similar whatever magnitudes above 17 for the g-band magnitude. It is a very interesting result because it means that the CNN performance does not depend on the magnitude but only on the number of objects in the training database. This effect is less visible on the right histogram of Fig. 5. Indeed, it is enough to consider only 5% of the training database to reach a recall between 98% and 99%. This experiment shows that the CNN is invariant to redshift.

In the testing base, for a fixed recall of 0.97, 175 new quasars detected by the CNN have never been identified before. We call them quasar candidates. Figure 6 represents the spatial distribution of found quasars in the testing base by the CNN. The red crosses characterize the new quasar candidates.

As we can see, the quasars detected by the CNN are distributed in an uniform manner. Figure 7 shows the average number of quasars in the sky per square degree, detected by the CNN, against the recall. As the recall increases, the number of quasars per square degree increases, which is consistent as we detected more and more quasars. For a recall around 0.92, the average number of quasars per square degree is about 20. Then, this number is drastically increased because the precision is reduced and the sample is contaminated by sources which are not quasars.

It is also interesting to highlight that a well known property of quasars is met by the new quasar candidates, namely a bluer when brighter tendency. This trend has been well established in the UV and optical color variations in quasar (e.g., Cristiani et al. 1997; Giveon et al. 1999; Vanden Berk et al. 2004). Figure 8 represents the amplitude of variations of detected quasars in the u-band filter against the r-band filter at different recalls. We note that 83.6%, and 88.7% of variation amplitudes in the u-band filter are larger than in the r-band filter for a recall of 0.90, and 0.97 respectively. Thus the detected quasars show larger variation amplitudes in bluer bands and so a strong wavelength dependence.

5.3. Comparison with a random forest classifier

We compare the performance of our algorithm with that of a random forest classifier whose we empirically estimated the best parameters on the same database. It contains 400 decision trees

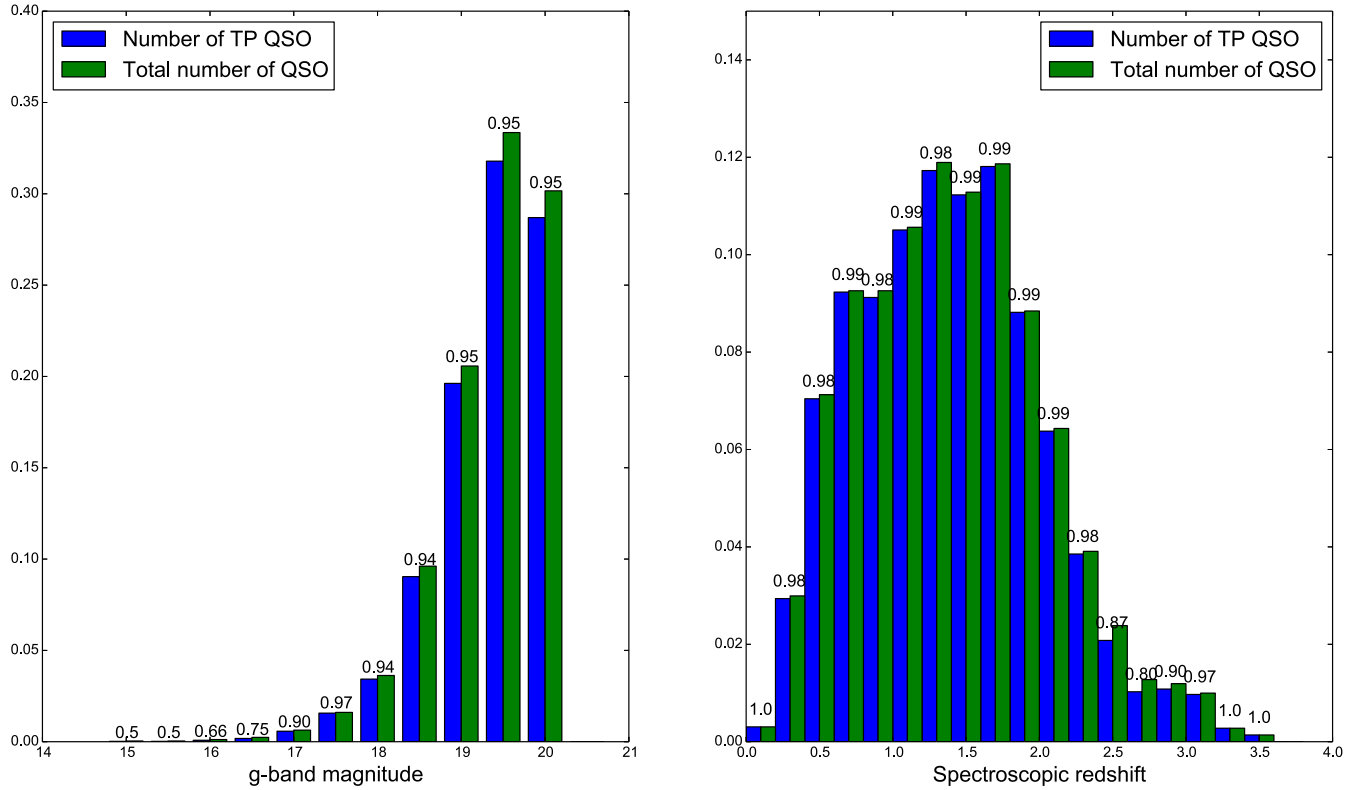

Fig. 5. *Left*: performance of the CNN depending on the median g-band magnitude. *Right*: performance of the CNN in function of the redshift. On each histogram the blue bars represent the number of well detected quasars by the CNN and the green bars the total number of quasars inside the corresponding bin. The recall is indicated over each couple of bars.

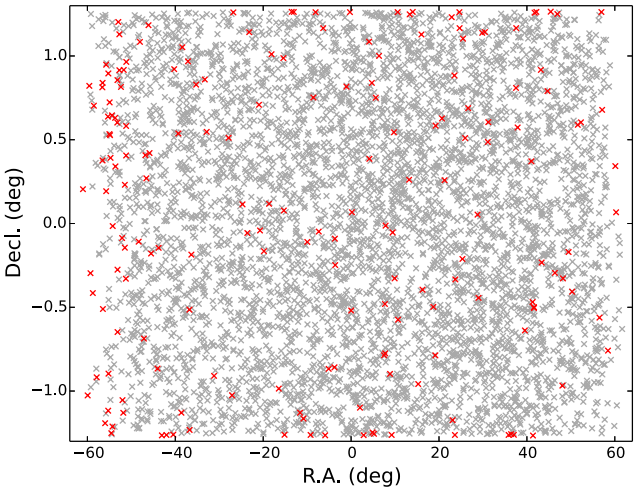

Fig. 6. Spatial distribution of quasars detected by the CNN during the testing phase. The gray crosses represent the well-known detected quasars, and the red crosses are the 175 new quasar candidates. They are uniformly distributed in the sky.

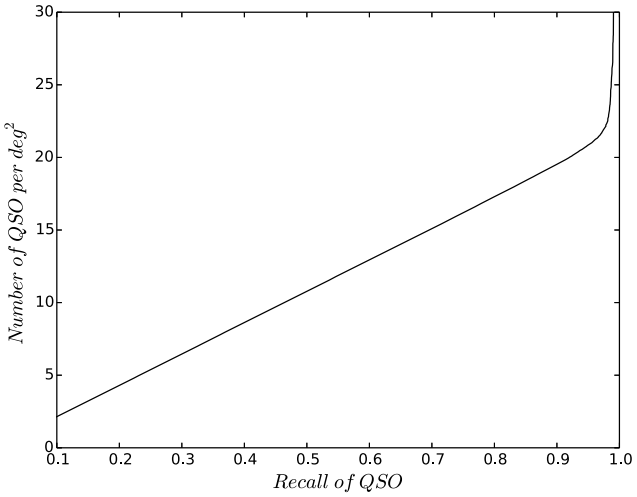

Fig. 7. Average number of quasars in the sky per square degree, detected by the CNN, against the recall. The larger the recall, the higher the number of the detected quasars is. For a recall of 0.92, the average number of detected quasars is 20 per square degree. Then this number is drastically increased since the precision is reduced and so the contamination of non-quasar sources is increased.

and an unlimited depth. The features used are included in a python library named FATS (Feature Analysis for Time Series; Nun et al. 2015) which is a compilation of some of the existing light-curve features.

At a fixed recall of 0.90, the precision is 0.988 for the CNN and 0.985 for the random forest. Then for a fixed recall of 0.97, the precision is 0.964, and 0.973 for the CNN and the random forest respectively. The performances of the two methods are closed and a little better for the random forest. A possible explication concerns the number of freedom degrees. Indeed for the random forest, about 640 000 parameters are defined whereas for the CNN there are about 13 000 000. Thus, due to the large number of parameters that have to be determined by backpropagation, the CNN should have better performance with more data, especially with large surveys.

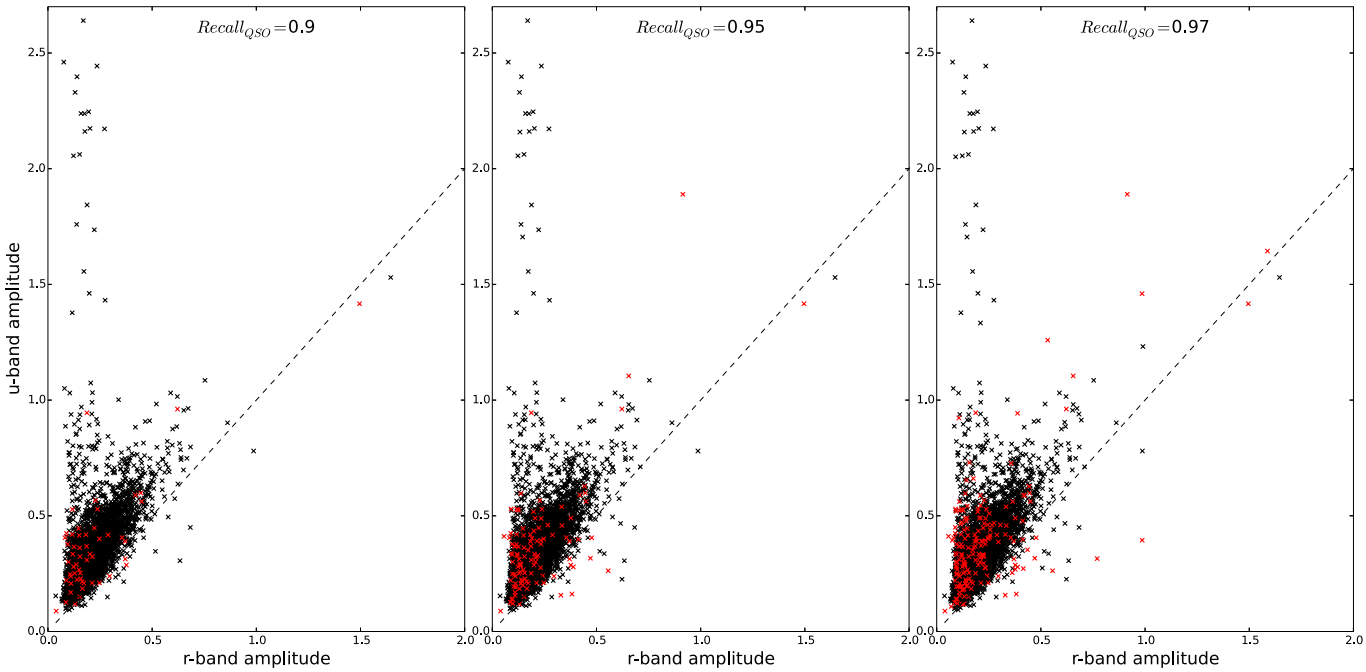

Fig. 8. Amplitudes of variation of quasars detected by the CNN from the u-band filter against those from the r-band filter at three different recalls: 0.90, 0.95, and 0.97. The black crosses represent all the known quasars during the testing phase. The red crosses are the 175 new quasar candidates. The dashed lines represent the line $y = x$. The quasars show larger variation amplitudes in bluer-bands. This tendency highlights a strong wavelength dependence.

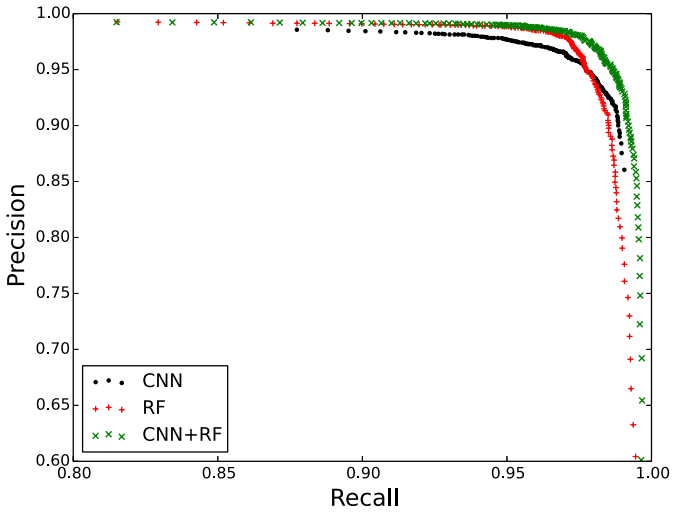

Fig. 9. ROC curve which plots the precision of the classifier against the recall. The performance of the CNN classifier is represented by black dots, those of the random forest by red plus and the performance of the combination of the two classifiers is represented by green crosses.

5.4. Combination of a CNN and a random forest

We combine the probabilities given by the CNN and the random forest by averaging them. For a fixed recall of 0.90 the precision is 0.99 and for a recall of 0.97, the precision is 0.98. Thus the combination of the two classifiers makes good performance even better. Figure 9 shows the receiver operating characteristic curve (ROC curve hereafter) which is a graphical plot that illustrates the performance of a classifier by plotting the precision against the recall. We can see that the random forest performance is better than that of the CNN until a recall of 0.978, where the CNN performance slightly drops. Moreover the ROC curve representing the combination of the two classifiers is above the two others and shows that combining a CNN classifier and a random forest classifier gives better classification performance.

The improvement obtained by the combination of the CNN and the random forest can be explained by the complementarity between the features given to the random forest and the features extracted by the CNN. Indeed, features used by the random forest are defined by the user and are specific for the classification of light curves of variable objects in general but they could not be perfectly designed for this classification problem that we considered. On the other hand, the CNN learns from scratch without any prior. The CNN found relevant features that are specific to the used database and so complete the information given by the features used by the RF. However, since the CNN learns features from the data, if there is not a large number of examples for a kind of objects, such as the high redshift quasars, the CNN does not find and learn the best features. In this case, it is relevant to use the results from the random forest to improve the classification. So depending on the number of data, the random forest features or the CNN features can complete each other.

6. Photometric redshifts of quasars

Photometric redshifts are a way to determine the redshift of an object by using only the apparent magnitudes through different filters, or photometric images. They constitute a powerful technique because they allow us to be free of spectroscopy data which are limited by the brightness of the source and by the cost of instruments. This methodology has been developed by Baum (1962) by observing the spectral energy distribution (SED) of six elliptic galaxies in the Virgo cluster in nine bands from 3730 Å to 9875 Å. The approach using template-fitting models which extracts features from celestial observational information and then matches them with the designed templates constructed by theoretical models or real observations has been used intensively (e.g., Bolzonella et al. 2000; Coupon et al. 2009; Ilbert et al.

Table 1. Evaluation of the KNN classifier efficiency using different features based on the absolute error and the error given by a χ^2 test.

Feature	Absolute error	χ^2	best K
Mean	0.282	0.239	2
Mean+error	0.283	0.240	2
Mean+color+error	0.263	0.199	3
Mean+color Amplitudes+error	0.253	0.182	4
Color+error	0.226	0.163	4
Color	0.226	0.156	6

Notes. K is the number of neighbors taken into account by the KNN algorithm.

2010). However the accuracy of the method strongly depends on simulated or real data. Moreover, the emergence of massive photometric data obtained by multiple large-scale sky surveys suggests the need of an automatic method such as machine learning algorithms. Several methods were used to estimate photometric redshifts of galaxies or quasars like a K-nearest neighbors (e.g., Zhang et al. 2013; Kügler et al. 2015), an artificial neural network (e.g., Firth et al. 2003; Collister & Lahav 2004; Blake et al. 2007; Oyaizu et al. 2008; Yèche et al. 2010; Zhang et al. 2009), both a K-nearest neighbors and a support vector machine (Han et al. 2016).

We propose to predict the photometric redshifts of quasars with a CNN. To do this, we used 80% of the quasar light curves for the training database and 20% for the testing database. To reduce the variability we cross validate the experiment and only show the mean of the results. The distribution of known spectroscopy redshifts are sliced in 60 bins of 0.04 in width. We used a network with a similar architecture that is represented in Fig. 4. The softmax gives the probability of belonging to each redshifts class. To predict the final regression value, results of each class are added by weighting them by the probability given by the Softmax. Again, the network takes the LCI as input (see Sect. 3.1) so as to include the information of the variability of objects in the estimation of redshifts.

To evaluate the proposed method, we compare it with a more classical approach using an extraction of features. For that, we compared the performances of four classifiers namely a K-nearest neighbors (KNN), a support vector machine (with linear and Gaussian kernels), a random forest and a Gaussian process classifier.

For each of these classifiers, we used the best combination of features among the mean of magnitudes, the magnitude errors, the amplitude of magnitudes, the colors and all characteristics included in the python library FATS. In the evaluation, the best results are obtained by using a KNN and only the color as a characteristic. Indeed as we can see in Table 1 a learning phase with only the color as a feature shows the lower absolute error of 0.226 and the lower residual of the χ^2 test with a value of 0.156. In this case, the number of neighbors taking into account, indicated by the number K in Table 1, is equal to six. Thus we use the performance of the KNN by extracting only the color to be compared to the performance of the CNN (see Table 1).

Figure 10 compares photometric redshifts predicted by the KNN (panel A) and the CNN (panel B) against the spectroscopic redshifts of around 9000 quasars. The color indicates the density of quasars in percentages. We note d_{CNN} and d_{KNN} the density of quasars in percentages given by the CNN and the KNN approaches respectively. Redder the color, higher the density of quasars is. For the two methods, the density of quasars is the highest on the line $y = x$, showing that the most of photometric redshifts are well estimated by the two classifiers. We remark that the density is the highest for redshifts below 2.5, since the database contains a small number of high redshifts, less than 10% which is then divided between the training and the testing databases.

Table 2. Comparisons of the accuracy and the dispersion obtained with the CNN, the KNN and the merge of the KNN and the CNN, by computing the percentages in different $|\Delta z|$ ranges and the root mean square (rms).

	$\lvert\Delta z\rvert < 0.1$ (%)	$\lvert\Delta z\rvert < 0.2$ (%)	$\lvert\Delta z\rvert < 0.3$ (%)	rms
CNN	79.32	86.64	91.69	0.352
KNN	73.72	82.46	90.09	0.395
KNN+CNN	80.43	87.07	91.75	0.349

Panel C in Fig. 10 compares the two approaches in the estimation of the photometric redshifts since it represents the difference between the density in percentages, d_{CNN} and d_{KNN} noted Δd. When the value of Δd is positive (shown on the plot), the density of quasars given by the CNN is higher than those obtained by the KNN. Contrariwise when the value of Δd is negative (also shown on the plot) the density of quasars given by the KNN is higher than those given by the CNN. We can see that the line $y = x$ appears in red color, so the values of Δd are positives showing that the density of quasars obtained by the CNN is higher than those given by the KNN and so that the CNN better predicts redshifts equals to spectroscopic redshifts than the KNN. On the contrary, the regions around the line $y = x$ are in blue meaning that the KNN has a higher error rate than the CNN and predict more catastrophic redshifts.

The better accuracy of the CNN is also visible if we compare the distribution of the absolute error in the estimation of photometric redshifts of the two methods (see Fig. 11). Indeed we can see that the histogram is narrower for the estimations of photometric redshifts made by the CNN than those made by the KNN. In addition the percentage of redshift estimations with an absolute error higher than 0.1, 0.2, and 0.3 are respectively of 38.03%, 24.06%, and 19.07% for the CNN; for the KNN they are of 45.78%, 30.04%, and 22.89%. Thus the number of catastrophic photometric redshifts is significantly reduced with the CNN.

We also define two quantities frequently used to evaluate accuracy and dispersion of the used method that are the percentages in different $|\Delta z|$ ranges defined as

$$\Delta z = \frac{z_{spec} - z_{phot}}{1 + z_{spec}} \tag{2}$$

and the rms of $|\Delta z|$ to test our redshifts prediction approach. The better accuracy of the CNN is confirmed (see Table 2) since the proportions of $|\Delta z|$ are equals to 78.09%, 86.15%, 91.2% for the CNN, against 73.72%, 82.46%, 90.09% for the KNN. This means that more photometric redshifts are estimated with a low error by the CNN than the KNN. In addition, the dispersion of photometric redshifts is also lower with the CNN, since the rms is 0.352 for the CNN against 0.395 for the KNN.

However the CNN has worse performance than the KNN for the prediction of redshifts higher than 2.5 (which is visible on Panel C in Fig. 10). It is due to the small number of redshifts higher than 2.5 in the database. There are only 600 quasars with

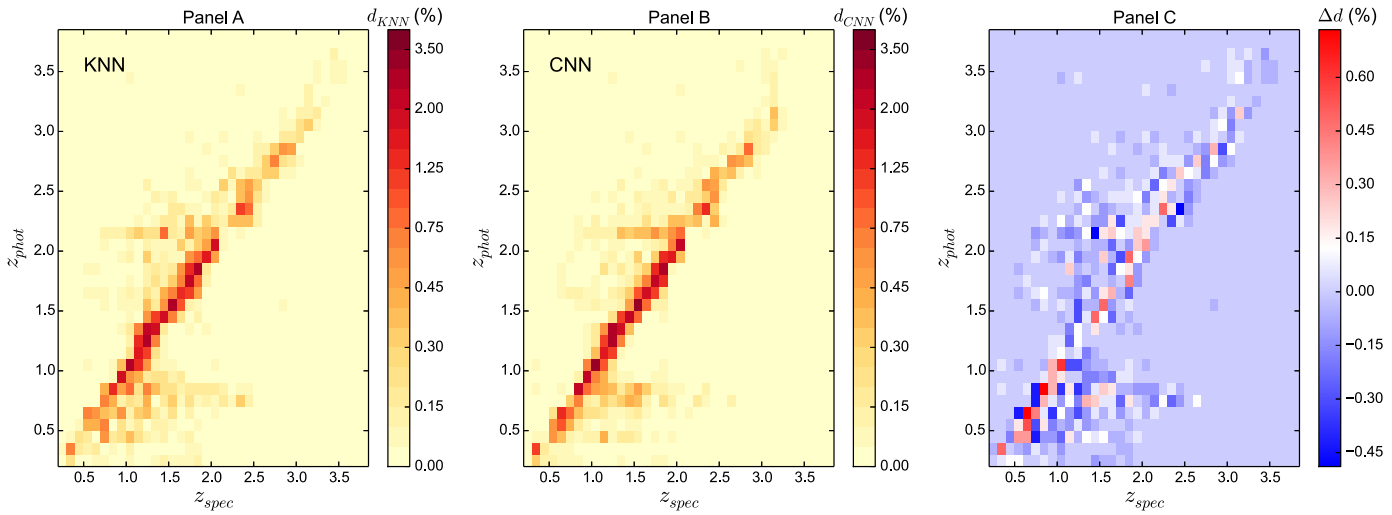

Fig. 10. *Panels A* and *B*: comparison of the photometric redshifts predicted by the KNN and the CNN respectively against the spectroscopic redshifts. The color indicates the density of quasars in percentages. Redder the color, higher the density of quasars is. The line $y = x$ is red which means that the density of quasars is the highest and so the two methods well estimate most of the photometric redshifts compared to spectroscopic redshifts. *Panel C*: difference in percentages between the density of quasars given by the CNN and the density of quasars obtained by the KNN, noted Δd. In other words, when Δd is positive (resp. negative), the color is red (resp. blue), and it means that the density of quasars given by the CNN (resp. KNN) is higher than those obtained by the KNN (resp. CNN).

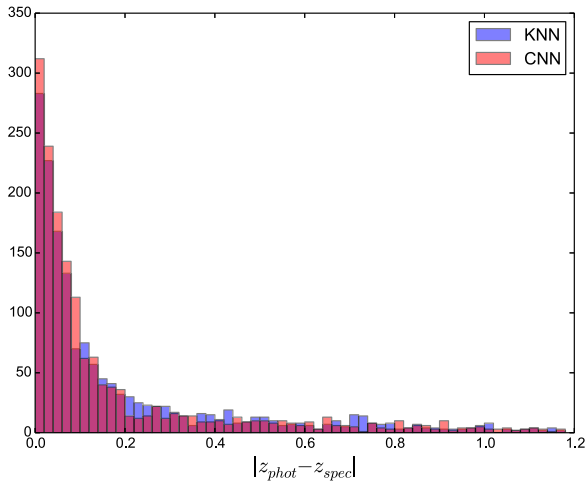

Fig. 11. Blue and red histograms representing the distribution of the absolute error for the estimation of photometric redshifts by using a KNN and a CNN classifiers respectively. The number of catastrophic redshifts is reduced with the CNN as the percentage of redshift estimation with an absolute error higher than 0.1 is about 45.78% for the KNN and 38.03% for the CNN.

$z_{spec} > 2.5$ in the learning database and the CNN needs a lot of examples to converge.

To solve this problem, after the training of the KNN and the CNN we combine these two approaches in a KNN+CNN architecture. The final prediction of the KNN+CNN architecture will depend on the redshift predicted by the KNN model. If the KNN predicts a redshift higher than 2.5, this prediction is used as the final prediction. Otherwise, the prediction given by the CNN is used as the final prediction. We note p_{KNN} and p_{CNN} the prediction given by the KNN and the CNN respectively. The prediction given by the KNN+CNN architecture, noted $p_{\mathrm{KNN+CNN}}$ is defined as

$$p_{\mathrm{KNN+CNN}} = \begin{cases} p_{\mathrm{KNN}} & \text{if } p_{\mathrm{KNN}} > 2.5 \\ p_{\mathrm{CNN}} & \text{otherwise} \end{cases} . \quad (3)$$

So, when the CNN does not have enough examples to learn a robust model, that is, for the high redshift estimations, the KNN model is used.

The performance given by the KNN+CNN architecture is a very interesting results as shown in Table 2. Indeed, the combination of the two classifiers reduces the number of catastrophic redshifts and the dispersion, since the proportions of $|\Delta z|$ are now equal to 80.43%, 87.07%, 91.75% and the value of rms is 0.349.

7. Conclusions

First, we have presented an original method based on a convolution neural network to classify and identify quasars in Stripe 82. The network takes the light curve images as input which are built from light curves of each object in the five *ugriz* filters, so as to include both the crucial information of the variabiliy and the colors in the learning of the network. The CNN classifier presents good results for the classification of quasars with a precision of 0.988 at a fixed recall of 0.90. For the same recall, the precision given by a random forest is 0.985. The very promising result is obtained by the combination of the CNN and the random forest giving precisions of 0.99 for a recall of 0.90.

Then, during the testing phase 175 new quasar candidates were detected by the CNN, with a fixed recall of 0.97. They are uniformly spatially distributed and they validate the bluer when brighter tendency.

Finally, we have used a CNN to predict the photometric redshifts of quasars. The performance of the CNN is higher than that of the KNN at redshifts below 2.5 with the best parameters determined experimentally. Indeed, the proportions of $|\Delta z|$ and rms error of predicted photometry redshifts are 78.09%, 86.15%, 91.2%, and 0.359 for the CNN; for the KNN they are 73.72%, 82.46%, 90.09%, and 0.395. The number of catastrophic redshift is also reduced by using a CNN, since the number of photometric redshifts with an absolute error higher than 0.1 is about 38.03% for the CNN against 45.78% for the KNN. Moreover the combination of a CNN and a KNN is a very promising method which better estimates redshifts higher than

2.5 and reduces the dispersion and the number of catastrophic redshifts.

Several improvements can be made for further studies. The most trivial is to use another catalog with a larger amount of data, because deep learning usually shows better results when there is more information. The second improvement consists of not dividing by averaging observations taken on two consecutive days, during the creation of the LCI. Indeed it is an approximation needed to reduce the computational cost, but it could be interesting to evaluate its impact on the results. Another interesting improvement is to take the errors into account in the learning phase which could show important information.

In conclusion, we wish to emphasis that the development of a method able to estimate well the photometric redshifts using only photometric information is essential for the future of big databases like LSST. Understanding that deep learning is more and more efficient as the size of the data increases, the future of this method is very promising.

References

Abazajian, K. N., Adelman-McCarthy, J. K., Agüeros, M. A., et al. 2009, ApJS, 182, 543
Baum, W. A. 1962, in Problems of Extra-Galactic Research, ed. G. C. McVittie, IAU Symp., 15, 390
Blake, C., Collister, A., Bridle, S., & Lahav, O. 2007, MNRAS, 374, 1527
Blomme, J., Sarro, L. M., O'Donovan, F. T., et al. 2011, MNRAS, 418, 96
Bolzonella, M., Miralles, J.-M., & Pelló, R. 2000, A&A, 363, 476
Collister, A. A. & Lahav, O. 2004, PASP, 116, 345
Cortes, C. & Vapnik, V. 1995, Mach. Learn., 20, 273
Coupon, J., Ilbert, O., Kilbinger, M., et al. 2009, A&A, 500, 981
Cristiani, S., Trentini, S., La Franca, F., & Andreani, P. 1997, A&A, 321, 123
Croom, S. M., Richards, G. T., Shanks, T., et al. 2009, MNRAS, 392, 19
Dubath, P., Rimoldini, L., Süveges, M., et al. 2011, MNRAS, 414, 2602
Duda, R. O. & Hart, P. E. 1973, Pattern classification and scene analysis (J. Wiley & Sons)
Eyer, L. & Blake, C. 2005, MNRAS, 358, 30
Firth, A. E., Lahav, O., & Somerville, R. S. 2003, MNRAS, 339, 1195
Frieman, J. A., Bassett, B., Becker, A., et al. 2008, AJ, 135, 338
Giveon, U., Maoz, D., Kaspi, S., Netzer, H., & Smith, P. S. 1999, MNRAS, 306, 637
Han, B., Ding, H.-P., Zhang, Y.-X., & Zhao, Y.-H. 2016, Res. Astron. Astrophys., 16, 074
He, K., Zhang, X., Ren, S., & Sun, J. 2015, in Proc. IEEE Int. Conf. Computer Vision (ICCV), ICCV '15 , 1026
Hernitschek, N., Schlafly, E. F., Sesar, B., et al. 2016, ApJ, 817, 73
Hopkins, P. F., Hernquist, L., Cox, T. J., et al. 2006, ApJS, 163, 1
Huertas-Company, M., Gravet, R., Cabrera-Vives, G., et al. 2015, ApJS, 221, 8
Ilbert, O., Salvato, M., Le Floc'h, E., et al. 2010, ApJ, 709, 644
Ioffe, S., & Szegedy, C. 2015, in Proceedings of the 32nd International Conference on Machine Learning (ICML-15), eds. D. Blei & F. Bach (JMLR Workshop and Conference Proceedings), 448
Ivezić, Ž., Smith, J. A., Miknaitis, G., et al. 2007, AJ, 134, 973
Jia, Y., Shelhamer, E., Donahue, J., et al. 2014, ArXiv e-prints [arXiv:1408.5093]
Joly, A., Goëau, H., Glotin, H., et al. 2016, in LifeCLEF 2016: Multimedia Life Species Identification Challenges, eds. N. Fuhr, P. Quaresma, T. Gonçalves, et al. (Cham: Springer International Publishing), 286
Krizhevsky, A., Sutskever, I., & Hinton, G. E. 2012, in Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3, Lake Tahoe, Nevada, United States, 1106
Kügler, S. D., Polsterer, K., & Hoecker, M. 2015, A&A, 576, A132
Le Guennec, A., Malinowski, S., & Tavenard, R. 2016, in ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, Riva Del Garda, Italy
Lopez, S., Barrientos, L. F., Lira, P., et al. 2008, ApJ, 679, 1144
LSST Science Collaboration 2009, ArXiv e-prints [arXiv:0912.0201]
Meusinger, H., Hinze, A., & de Hoon, A. 2011, A&A, 525, A37
Nair, V., & Hinton, G. E. 2010, in Proceedings of the 27th International Conference on Machine Learning (ICML-10), eds. J. Fürnkranz & T. Joachims (Omnipress), 807
Nun, I., Protopapas, P., Sim, B., et al. 2015, ArXiv e-prints [arXiv:1506.00010]
Oyaizu, H., Lima, M., Cunha, C. E., et al. 2008, ApJ, 674, 768
Peng, N., Zhang, Y., Zhao, Y., & Wu, X.-b. 2012, MNRAS, 425, 2599
Peters, C. M., Richards, G. T., Myers, A. D., et al. 2015, ApJ, 811, 95
Portinari, L., Kotilainen, J., Falomo, R., & Decarli, R. 2012, MNRAS, 420, 732
Quinlan, J. R. 1986, Mach. Learn., 1, 81
Rimoldini, L., Dubath, P., Süveges, M., et al. 2012, MNRAS, 427, 2917
Rumelhart, D. E., Hinton, G. E., & Williams, R. J. 1986, in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1, (Cambridge, MA, USA: MIT Press), 318
Russakovsky, O., Deng, J., Su, H., et al. 2015, Int. J. Comput. Vis. (IJCV), 115, 211
Schneider, D. P., Richards, G. T., Hall, P. B., et al. 2010, AJ, 139, 2360
Sesar, B., Ivezić, Ž., Lupton, R. H., et al. 2007, AJ, 134, 2236
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. 2014, J. Mach. Learn. Res., 15, 1929
Szegedy, C., Liu, W., Jia, Y., et al. 2015, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1
The Dark Energy Survey Collaboration 2005, ArXiv e-prints [arXiv:astro-ph/0510346]
Vanden Berk, D. E., Wilhite, B. C., Kron, R. G., et al. 2004, ApJ, 601, 692
Yèche, C., Petitjean, P., Rich, J., et al. 2010, A&A, 523, A14
York, D. G., Adelman, J., Anderson, Jr. J. E., et al. 2000, AJ, 120, 1579
Zhang, Y., Li, L., & Zhao, Y. 2009, MNRAS, 392, 233
Zhang, Y., Ma, H., Peng, N., Zhao, Y., & Wu, X.-b. 2013, AJ, 146, 22

Appendix A

Table A.1. Characteristics of each layer of the CNN architecture.

Layers	Inputs	Kernel size	$h \times w$	#Feature maps
P1, P2	LCI	5×1, 11×1 (stride 2)	850×5	1, 1
TC1, TC2, TC3	P1	11×1, 21×1, 41×1	850×5	16 16, 16
TC4, TC5, TC6	P2	11×1, 21×1, 41×1	850×5	16, 16, 16
MC1	C1	1×5	850×1	96
TC7, TC8	MC1 and C1	11×1, 21×1	850×5 or 850×1	24, 24
P3	C2	3×1 (stride 2)	425×1	48
P4	C3	3×1 (stride 2)	425×5	48
P5	LCI	21×1 (stride 4)	425×5	1
TC9, TC10, TC11	P5	5×1, 11×1, 21×1	425×5	16, 16, 16
TC12, TC13	C4	11×1, 21×1	425×5	24, 24
MC2	C5	1×5	425×1	48
TC14	C6 and C7	11×1	425×1 or 425×5	96, 96
P6	TC14	3×1 (stride 2)	212×1	96
P7	TC14	3×1 (stride 2)	212×5	96
TC15	P6 and P7	11×1	425×1 or 425×5	48, 48
P8	TC15	3×1 (stride 2)	106×1	48
P9	TC15	3×1 (stride 2)	106×5	48
P10	LCI	41×1 (stride 16)	106×5	1
TC16, TC17, TC18	P10	5×1, 11×1, 21×1	106×5	16, 16, 16
TC19	C8	11×1	106×5	48
MC3	TC19	1×5	106×1	48
TC20	C9 and C10	11×1	106×1 or 106×5	128, 128
P11	TC20	3×1 (stride 2)	53×1	48
P12	TC20	3×1 (stride 2)	53×5	48
P13	LCI	61×1 (stride 32)	53×5	1
TC21, TC22, TC23	P13	5×1, 11×1, 21×1	53×5	16, 16, 16
TC24	C11	11×1	53×5	64
TC25, TC26	C12	11×1, 21×1	53×5	64, 64
MC4	C13	1×5	53×1	64
TC27, TC28, TC29, TC30	C14	5×1, 11×1, 21×1, 41×1	53×1	48, 48, 48, 48
FC1, FC2	C15, FC1	–	–	1024, 1024

Notes. The table lists: the name of the layer, the input layer, the size of the convolution kernel (in pixels), the size in pixels (height $\times$ width) of resulting feature maps and the number of resulting feature maps. The concatenation layers are not represented here but they are present in Fig. 4.