

A CNN-RNN Framework for Image Annotation from Visual Cues and Social Network Metadata

Tobia Tesan
Quantexa Ltd, London, UK
tobiatesan@quantexa.com

Pasquale Coscia
University of Padova, Italy
pasquale.coscia@unipd.it

Lamberto Ballan
University of Padova, Italy
lamberto.ballan@unipd.it

Abstract—Images represent a commonly used form of visual communication among people. Nevertheless, image classification may be a challenging task when dealing with unclear or non-common images needing more context to be correctly annotated. Metadata accompanying images on social-media represent an ideal source of additional information for retrieving proper neighborhoods easing image annotation task. To this end, we blend visual features extracted from neighbors and their metadata to jointly leverage context and visual cues. Our models use multiple semantic embeddings to achieve the dual objective of being robust to vocabulary changes between train and test sets and decoupling the architecture from the low-level metadata representation. Convolutional and recurrent neural networks (CNNs-RNNs) are jointly adopted to infer similarity among neighbors and query images. We perform comprehensive experiments on the NUS-WIDE dataset showing that our models outperform state-of-the-art architectures based on images and metadata, and decrease both sensory and semantic gaps to better annotate images.

I. INTRODUCTION

Images represent an effective and immediate form of expression commonly used to share events and moments of our daily lives. This is particularly true nowadays with the rising popularity of social networks such as Facebook, Twitter and Instagram. Additional information like similar images and social network *metadata*, are often employed to provide external context and to emphasize moods and messages. Dealing with such contextual data could advantage visual recognition tasks, such as image tagging and retrieval [1], in ambiguous cases where main parts are occluded or unrecognizable (as in Figure 1). In this paper we build on the intuition that a context of additional weakly-annotated images can help in disambiguating the visual classification task, as shown in the seminal work by Johnson *et al.* [2].

The idea of using contextual data to improve visual recognition is not new [3], [4]. Even humans usually benefit from the context in object detection and scene recognition [5]. In particular, in this work we exploit the (noisy) contextual information given by metadata embedded in images shared on social-networks. Metadata could be very useful to classify examples that occur very rarely or showing visual elements in non-prototypical views. Here image and network metadata can be considerably effective in bridging the sensory and the semantic gap [6], [7].

Various types of metadata are shared on social-networks. For example, digital photos normally provide information like ISO, exposure, location or timestamp. Users may also add

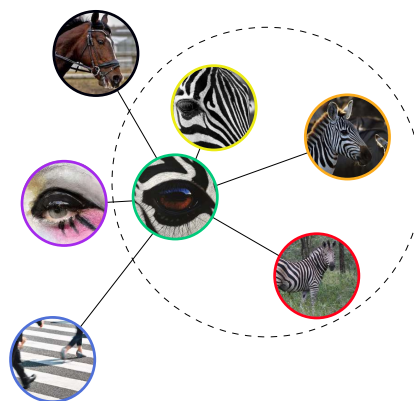


Fig. 1: Some images might be hard to recognize without additional context. However, related images on a social network typically share similar *metadata*. Based on this intuition, given an image, we retrieve a neighborhood of images sharing similar metadata (e.g. tags) to assist the image annotation task. Our approach builds on [2] and introduces more advanced semantic mapping and CNN-RNN fusion schemes.

textual descriptions, or provide names of people which appear in photos. Several works have exploited metadata to improve image classification and retrieval, mostly using user-generated tags [8], [9], [10], [11], GPS data [12], [13] or groups [14]. In [2], image metadata such as tags or Flickr groups are used nonparametrically to generate a pool of related images, that can be further exploited by a deep neural network to blend visual information from a given image and its neighborhood. The key contribution of the approach is a model that can deal with different metadata and adapts over time with no (or very limited) re-training. Thus the model reported state-of-the-art results on multilabel image annotation by taking advantage of strong visual models [15], [16] and flexible nonparametric approaches [17], [18].

In this work we explore different architectures based on both visual cues and external data (e.g., tags) to improve the simple fusion scheme presented in [2]. More specifically, we first focus on preserving distance between a test image x and its neighbors to capture more relevant labels, as well as on handling vocabulary changes when new terms are included. To this end, our proposed architectures attempt to better encode

the semantic meaning of tags through word embeddings [19], [20]. Second, we investigate and design different architectures for image-to-neighborhood features fusion. Here the main source of inspiration is given by recent CNN-RNN models for image classification and captioning [21], [22]. In these works, a CNN is used to extract the image feature vector, which is then fed into an RNN that either decodes it into a list of labels (multilabel image classification) or a sequence of words composing a sentence (captioning). In contrast, we investigate different strategies in which an RNN is used to sequentially blend the visual or multimodal information in a joint feature space.

The remainder of the paper is organized as follows. In Section II, we review related work in the area of image classification in a (noisy) multimodal scenario. In Section III, we present our deep network framework. We evaluate the performance of our method on the NUS-WIDE dataset [23], and Section IV shows that the approach improves previous state-of-the-art models [2], [21].

II. RELATED WORK

A. Image Tagging and Retrieval

The idea of harvesting images from the web to train visual classification models has been explored many times in the past [24], [25], [26]. Despite its simplicity, a popular and quite effective approach for automatic image annotation, that has been often used in early works, is nearest-neighbors based label transfer [27], [17]. More recently, deep networks have been applied extensively also in this domain achieving state-of-the-art results on many popular benchmarks [15], [16].

Among the vast literature on image tagging and retrieval [1], our work is mostly related to multimodal representation learning of images and labels. To this end, early works often model the association between visual data and labels in a generative way or rely on mapping images and labels to a common semantic space using techniques such as CCA or KCCA [28], [9], [29]. Hu *et al.* [30] observe that diverse levels of visual categorization are possible depending on the level of desired abstraction. Thus, they rely on structured inference to capture relationships among concepts in neural networks. In general, these approaches demonstrate the benefit of exploiting side information and correlations between visual features and labels, but they only rely on ground truth annotations.

B. Automatic Image Annotation with Metadata

Several previous works tackled the automatic image annotation task using social-network metadata [6], [12], [7], [14]. User-generated tags are significantly the most commonly used metadata for multilabel image classification. In [31], Guillaumin *et al.* consider a scenario in which only visual data is used at test time, but metadata from social media websites (such as Flickr) are available at training time and can be leveraged to improve classification using semi-supervised learning. Moreover, a combination of simple nonparametric models and metric learning is used in [8], while [18] focuses on selecting a better set of training images to drive the label

transfer. Flickr groups are exploited in [14] to derive a measure of image similarity which can encode broader correlations than user-generated tags and labels. A graph over tags, groups or common GPS location is used by Niu *et al.* [11] to define a semi-supervised topic model for image classification.

Our work falls in this area. Inspired by the model presented by Johnson *et al.* [2], we also use a deep network to blend the visual information extracted from a neighborhood of images sharing similar metadata. This idea has been also recently followed in [32] where a co-attention mechanism is used to construct a graph in which each node represents a relevant neighbor and correlated images are connected by edges. Our method differs from these works because we focus on defining a more effective architecture to combine visual cues and social-network metadata from both the test image and the neighborhood.

III. OUR FRAMEWORK

Our goal is to annotate images using side information carried by their neighbors. More specifically, we jointly exploit visual features as well as tags which commonly accompany images on social networks. Tags are embedded using different semantic mappings. Our models are built upon the work presented by Johnson *et al.* [2], where metadata are only used to retrieve similar images and the annotation task mainly relies on visual features. We propose two general architectures for images annotation, both based on visual features and image metadata (see Figure 2). Whereas visual models only exploit visual cues, joint models handle metadata which are directly fed to the neural network after a transformation step.

All the models generate nonparametrically a neighborhood Z_x for a query image x using metadata and then the networks are trained to classify x given its neighbors in Z_x . The neighborhood generation process is parametrized over a neighborhood size m and a max rank M . More specifically, let Z_x be the M -nearest neighbors of x according to a distance measure δ . The set of candidate neighborhoods for an image x is the set:

$$Z_x = \{s \in \mathcal{P}(X) : |s| = m\}, \quad (1)$$

where $\mathcal{P}(X)$ denotes the power set of X , that is the set of considered images. The prediction $s(x, \theta)$ is the average of $f(x, \vec{z}; \theta)$ over all candidate neighborhoods:

$$s(x, \theta) = \frac{1}{|Z_x|} \sum_{z \in Z_x} f(x, \vec{z}; \theta), \quad (2)$$

where x is the image to be classified, $\vec{z} = (z_1, z_2, \dots, z_m)$ are the neighbors and $f(x, \vec{z}; \theta)$ is the output of the neural network which takes into account their visual cues.

The model is trained by computing a loss function \mathcal{L} and minimizing:

$$\theta^* = \arg \min_{\theta} \sum_{(x,y) \in D_{train}} \mathcal{L}(s(x, \theta), y), \quad (3)$$

where y represent a subset of all possible labels that appear in D . Note that neighbors are ordered according to their distance

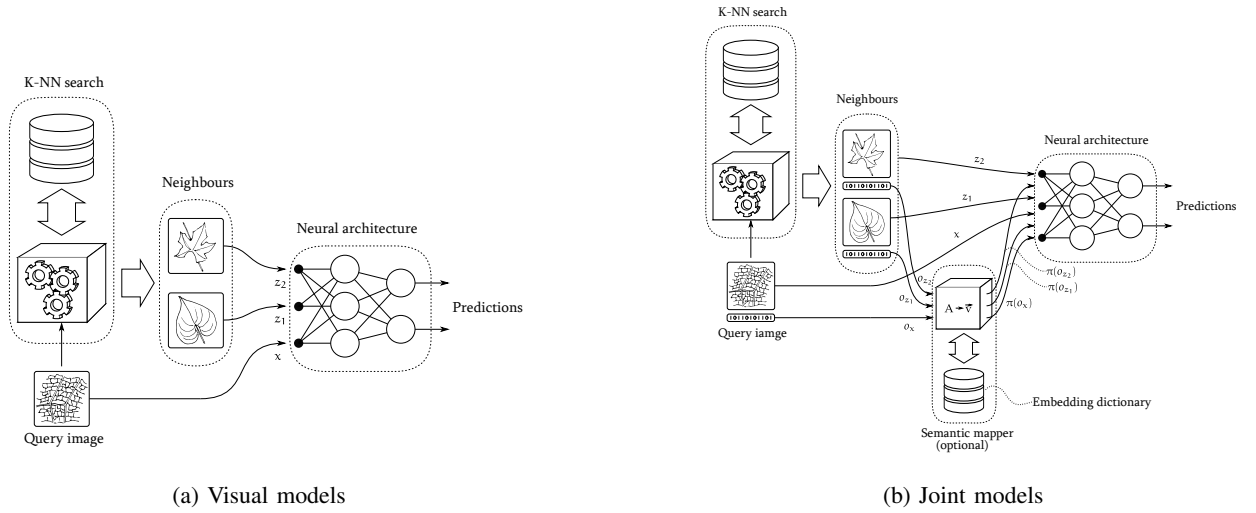


Fig. 2: General architectures of the proposed models. K-NN is used to retrieve similar images using metadata, while a neural network processes the retrieved information. (a) shows the architecture for visual models (as in [2]) where only visual features for both query image and neighbors are used to predict labels. (b) shows the architecture for joint models where metadata are also fed, possibly after a transformation step, to the final classification layer.

when fed to the neural network and thus the network may learn to treat the closest ones differently.

Joint models differ from visual models in that they enrich image representation with additional information. More specifically, such models use metadata which are directly fed to the final layer of the network after a transformation step $\pi(\cdot)$ involving a lookup in a dictionary of semantic embeddings. In this case, the prediction $s(x, \theta)$ is the average of $f(x, \pi(o_x), \bar{z}, \pi(\bar{o}_z); \theta)$, where o_x is the metadata vector for image x while $\pi(o_x)$ is its transform. We shall use $\pi(\bar{o}_z)$ as shorthand for $\text{map}(\pi, \bar{o}_z) = (\pi(o_{z_1}), \pi(o_{z_2}), \dots, \pi(o_{z_m}))$, where \bar{o}_z are metadata vectors for the neighborhood.

A. Metadata Encoding

Metadata representation may affect network's ability to recover correct annotations. For this reason, we firstly encode metadata without associating any meaningful representation to each word, i.e., semantically close words could be associated to distant vectors, and secondly consider more powerful word encoding techniques.

1) *One-hot Encoding*: We focus on social-network tags represented as binary vectors $o_x \in \{0, 1\}^\tau$. More specifically, let x the query image and $(t_{(1)}, t_{(2)}, \dots, t_{(n)})$ all relevant tags for x chosen from a vocabulary of τ tags, the binary vector o_x is the sum of the one-hot vectors for each of its tags:

$$o_x = \sum_{i \text{ s.t. } t_i \in \{t_{(1)}, t_{(2)}, \dots, t_{(n)}\}} e_i^\tau. \quad (4)$$

Using `id`, i.e., raw binary vectors, neighborhoods are computed using the Jaccard distance \mathcal{J} between binary vectors. Binary vectors o_x for each image x (or neighbor z_i) are directly

handled by the neural network, without further processing. The Jaccard distance is defined as follows:

$$\mathcal{J}(x, x') = 1 - \frac{|t_x \cap t_{x'}|}{|t_x \cup t_{x'}|} \quad (5)$$

with $\mathcal{J}(x, x) = 0$.

2) *Semantic-aware Encoding*: We also explore more powerful word embedding techniques in order to encode similar word into similar vectors. We consider a transformation that maps a vector o_x to a *semantic space* $\pi : \{0, 1\}^\tau \rightarrow \mathbb{R}^n$. It is clear that, unlike visual models, where metadata are used implicitly, a neural network trained to make predictions as a function of one or more binary vectors becomes useless if the vocabulary changes. Semantic maps π can decouple the low-level bit representation from the semantic meaning, making models learned on a tag vocabulary applicable to a different one, as long as an appropriate $\tilde{\pi}$ is available that maps the *new* binary vectors onto the *old* semantic space. More specifically, given a map or dictionary of embeddings $\beta : TAGS \rightarrow \mathbb{R}^n$ for some n , we define $\rho(o_x; \beta)$ as the sum of the vectors $\beta(t_{(i)})$ for each tag $t_{(i)}$ relevant for image x , i.e.:

$$\rho(o_x; \beta) = \sum_{i=1}^{\tau} o_{x_{(i)}} \cdot \beta(t_{(i)}). \quad (6)$$

For $\pi(x) = \rho(o_x; \beta)$, we consider two semantic embeddings. Firstly, we use a dictionary of `word2vec` embeddings [19]; they are obtained by training on a 100-billion-words subset of the Google News database and contain 300-dimensional vectors for 3 million words and phrases. We expect to recover some semantic information from the tags and improve performance, as well as achieving decoupling from

the low-level binary representation for joint architectures. We choose cosine distance for δ , defined as:

$$\text{sim}_{\text{cos}}(x_1, x_2) = 1 - \frac{\vec{x}_1 \cdot \vec{x}_2}{|\vec{x}_1| |\vec{x}_2|}. \quad (7)$$

Secondly, we use `WordNet` embeddings which works in the same fashion as `word2vec`, except that β is extracted from a dictionary where vector representations are optimized to be similar if the words are close on the `WordNet` taxonomy. Cosine distance is again our choice for δ . `WordNet` embeddings [20] comprise a dictionary of 650-dimensional vectors obtained from Princeton `WordNet 3.0`¹ with 60,000 words.

B. Visual Models

Visual models only rely on extracted visual features of input images without considering additional information. We consider three visual models based on fully-connected and recurrent layers.

1) *Visual-only*: This architecture acts as baseline; it simply amounts to a fully-connected layer over visual features $\phi(x)$ output by a CNN for an image x . Therefore,

$$f(x, \vec{z}; \theta) = W_y \Phi(x) + b_y \quad (8)$$

Note that \vec{z} is not used.

2) *LTN*: This is the model proposed in [2]. The label scores are computed as follows:

$$f(x, \vec{z}; \theta) = W_y \begin{bmatrix} v_x \\ v_z \end{bmatrix} + b_y \quad (9)$$

where $\vec{z} = (z_1, z_2, \dots, z_m)$ is a vector of neighbors obtained nonparametrically, x is the image to be classified, and

$$v_x = \sigma(W_x \Phi(x) + b_x), \quad (10)$$

$$v_z = \max_{i=1, \dots, m} (\sigma(W_z \Phi(z_i) + b_z)) \quad (11)$$

where σ is a ReLU activation function. The model is depicted in Figure 3a. Note that the weights W_z and b_z are shared among all (z_1, z_2, \dots, z_m) and $v_x, v_z \in \mathbb{R}^h$.

3) *RTN*: This architecture extends *LTN* by replacing the max-pooling operation with a RNN in order to better discriminate individual neighbors. Sequential image processing may allow the network to retain only relevant features of the neighborhood handling each image separately.

More specifically, the hidden state v_z is defined as follows:

$$v_z = \text{RNN}((z_1, z_2, \dots, z_m); W_{\text{RNN}}), \quad (12)$$

where the notation $\text{RNN}((i_1, \dots, i_n), W)$ denotes a recurrent neural network sequentially fed with inputs (i_1, \dots, i_n) while W are the corresponding parameters. In this case, RNN is a long short-term memory (LSTM) network with linear activation function. The other parameters remain unchanged. The model is depicted in Figure 3b.

C. Joint Models

Joint models are directly fed with metadata instead of leveraging metadata only implicitly along with visual features. Metadata improve the semantic level detected by extracted visual features. In the following, we define several architectures handling metadata (or their embeddings) using linear and recurrent layers.

1) *LTN+Vecs*: This architecture makes use of metadata o_x , i.e., metadata of image to be classified, which are concatenated to the output of the CNN of image x .

The output of the network is defined as follows:

$$f(x, \pi(o_x), \vec{z}; \theta) = W_y \begin{bmatrix} v_x \\ v_z \end{bmatrix} + b_y, \quad (13)$$

where

$$v_x = \sigma \left(W_x \begin{bmatrix} \Phi(x) \\ \pi(o_x) \end{bmatrix} + b_x \right). \quad (14)$$

v_z is defined as in *LTN* visual model. Note that such model does not use neighbor metadata vectors and it only relies on visual features of closest images. A transformation step is then applied to map metadata onto a new space (see Figure 3c).

2) *LTN+AllVecs*: This architecture, unlike the previous one, uses metadata vectors o_x of the image to be classified and metadata of its neighbors \vec{o}_z .

The output is defined as follows:

$$f(x, \pi(o_x), \vec{z}, \pi(\vec{o}_z); \theta) = W_y \begin{bmatrix} v_x \\ v_z \end{bmatrix} + b_y, \quad (15)$$

where v_x is defined as above and

$$v_z = \max_{i=1, \dots, m} \sigma \left(W_z \begin{bmatrix} \Phi(z_i) \\ \pi(o_{z_i}) \end{bmatrix} \right). \quad (16)$$

In this case, σ is a ReLU activation function. The model is depicted in Figure 3d.

3) *LTwin*: Unlike *LTN+AllVecs*, such architecture processes features and metadata using two separate pipelines, i.e., metadata are not concatenated with the images features. The neighbors are blended with a max-pooling layer, so the model is not able to discriminate between nearest and farthest neighbors.

The output of the network is defined as follows:

$$f(x, \pi(o_x), \vec{z}, \pi(\vec{o}_z); \theta) = W_y \begin{bmatrix} v_x \\ v_z \\ u_x \\ u_z \end{bmatrix} + b_y, \quad (17)$$

where v_x and v_z are defined as in the *LTN* model, while $u_x = \sigma(W_{x_u} \pi(o_x) + b_{x_u})$ and $u_z = \max_{i=1, \dots, m} \sigma(W_{z_u} \pi(o_{z_i}) + b_{z_u})$. Max-pooling is applied on both neighbors' features and their metadata. The model is depicted in Figure 3e.

4) *LTwin+RNN*: Unlike the previous architecture, such model replaces max-pooling layers with RNN networks to handle the neighbors. Once again, RNN is an LSTM with linear activation. The output is equal to *LTwin* architecture with $v_z = \text{RNN}((FC_{z_1}, \dots, FC_{z_m}); W_{\text{RNN}})$ and $u_z = \text{RNN}((FC_{o_{z_1}}, \dots, FC_{o_{z_m}}); W_{\text{ORNN}})$, where $FC_{(\cdot)}$ are outputs of fully-connected layers applied to image features and metadata, respectively. The model is depicted in Figure 3f.

¹<https://github.com/nlx-group/WordNetEmbeddings>

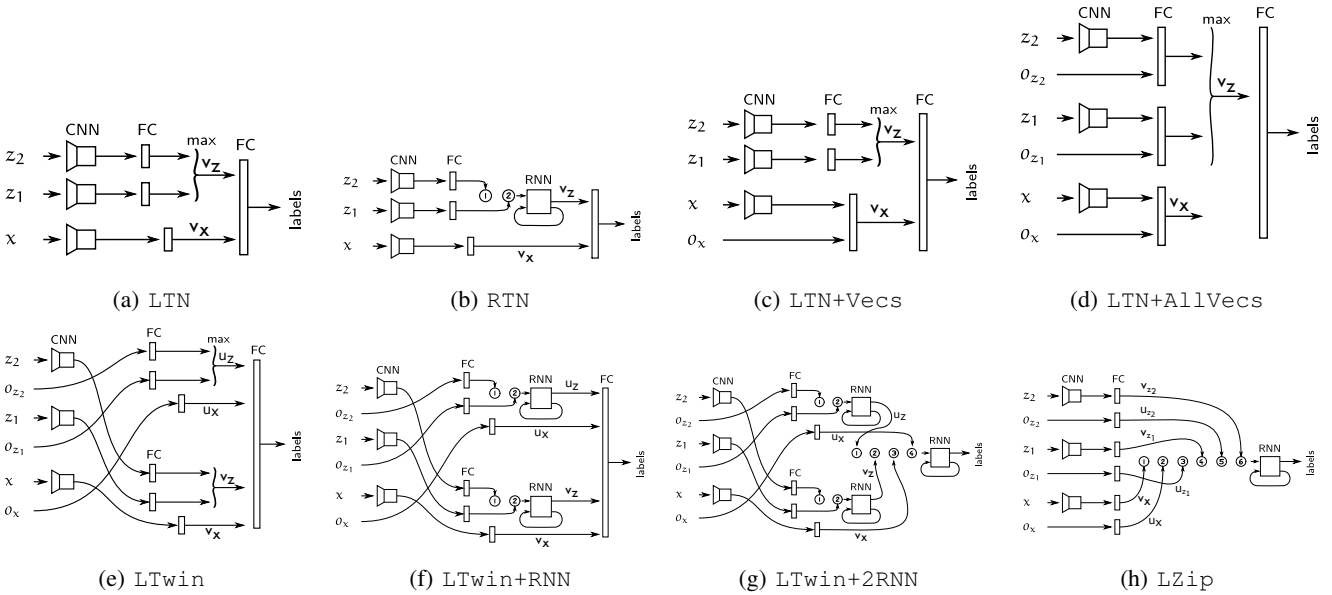


Fig. 3: Our architectures which leverage image features along with metadata. (a),(b) represent two visual models where metadata are not employed. (c)–(h) are different ways to fuse image features and metadata. In this work, as metadata we only use tags and we exploit recurrent layers and semantic embeddings in order to leverage contextual information.

5) *LTwin+2RNN*: This architecture differs from the previous one in that the final fully connected layer is also replaced with a RNN. The output is defined as follows:

$$f(x, \pi(o_x), \vec{z}, \pi(\vec{o}_z); \theta) = RNN((v_x, v_z, u_x, u_z); W_{f_{RNN}}), \quad (18)$$

where v_x, v_z, u_x and u_z are defined as in *LTwin+RNN*. The model is depicted in Figure 3g.

6) *LZip*: Finally, this architecture uses just one RNN to combine features and metadata which are separately processed by FC layers. The output is defined as follows:

$$f(x, \pi(o_x), \vec{z}, \pi(\vec{o}_z); \theta) = RNN((v_x, u_x, v_{z_1}, u_{z_1}, \dots, v_{z_m}, u_{z_m}); W_{RNN}). \quad (19)$$

The model is depicted in Figure 3h.

D. Implementation Details

We use RMSProp algorithm with He-Zhang initialization [33] and apply dropout with $p = 0.5$. We also set batch size dimension to 64 (in lieu of 50, as found in [2]) and $h = 500$. We apply L_2 regularization with $\lambda = 3 \times 10^{-4}$ and use a learning rate of 1×10^{-4} . λ was chosen with grid search. We use early stopping with a maximum of 10 and a minimum of 3 epochs, incremented to 15 and 5 for joint models, respectively. We run experiments with (3, 6), (6, 12) and (12, 24) as choices of (m, M) . Our CNN is the ImageNet pre-trained AlexNet [15] model available on Caffe, as in [2].

IV. EXPERIMENTS

1) *Dataset*: We use the NUS-WIDE dataset [23] which comprises 269,648 images uploaded on the photo sharing website Flickr, annotated with 81 ground truth labels for

evaluation. NUS-WIDE is highly unbalanced over classes, whereas the tag *sky* is relevant for around 53,000 images, many classes have less than a thousand images. We restrict ourselves to the fixed subset of 190,253 images used in [2], [32] for ease of comparison. The dataset comprises 422,364 unique Flickr tags, which we narrow down to the $\tau = 5000$ most frequent tags. The dataset is randomly partitioned to form training, validation and test sets of 110,000, 40,000 and 40,253 images, respectively. We average the results over 5 of such splits.

2) *Metrics*: We report per-label and per-image mean Average Precision (mAP), as well as precision and recall. Note that, in this area, the most common evaluation protocol assumes that an algorithm should assign a fixed number k of labels to each image. To this end, following prior work [16], [2], [21], we report results for $k = 3$. Since on NUS-WIDE the average number of labels per image is approx. 2.4, by assigning exactly 3 labels, no classifier can achieve unit precision and recall (thus we report on Table I the real upper bound for each metric). However, as also highlighted in [8], [2], [1], mAP directly measures ranking quality, so it naturally handles multiple labels and does not require to set a fixed number k . Therefore, mAP is the primary evaluation metric used further on in our evaluation.

A. Experimental Results

Table I shows our best results in comparison to several baselines and state-of-the-art models. First of all, the *LTwin* model outperforms the other methods on both mAP metrics. It is also important to note that for the corresponding models proposed in [2], our implementation of *LTN* achieves comparable results while *LTN+Vecs* has worse performance. Therefore,

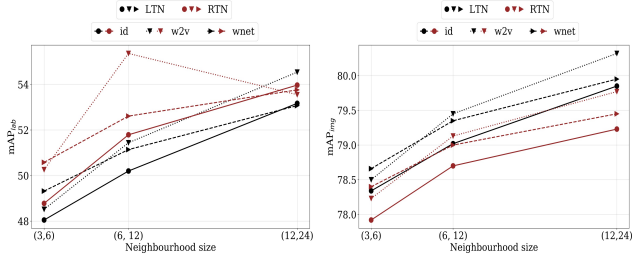


Fig. 4: mAP_{lab} and mAP_{img} for visual models varying the neighborhood size and semantic mapping to retrieve the neighbors. Black color refers to LTN model while the red one to RTN model. All the models outperform the visual-only baseline.

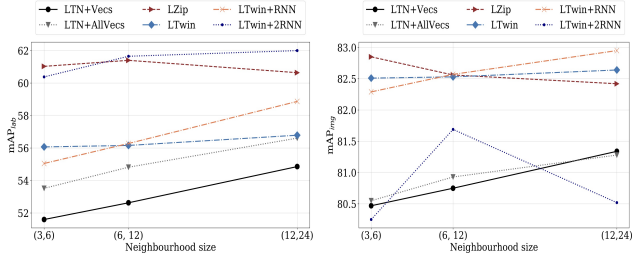


Fig. 5: mAP_{lab} and mAP_{img} for joint models varying the neighborhood size considering $\pi = id$ both for neighbors retrieval and metadata embedding.

the LTwin model achieves best results showing a 10 and 2 percentage performance increase on both mAP metrics w.r.t. the corresponding LTN+Vecs baseline.

More detailed results about all the different architectures presented in Section III are reported in Table II and Table III (all the results refer to a neighborhood size of (12, 24), highlighting a vast range of different combinations of architectures and encodings. We choose to focus our attention on mAP_{lab} and mAP_{img} since they better summarize classification performances. In general, we note that mAP_{lab} is the metric that is affected the most, whereas mAP_{img} remains more stationary.

1) *Visual Models*: As shown in Figure 4, for the same neighborhood, RTN leads to an improvement of mAP_{lab} of around 0.7 to 1.2 percentage points over LTN, in exchange for a drop of 0.2 to 0.4 percentage points of mAP_{img} . More interestingly, the gap between $\pi = id$ and word2vec is larger for RTN at low values of m . Notice how RTN with word2vec embeddings and a (3,6) neighborhood outperforms vanilla LTN with (6,12) neighborhood in terms of mAP_{lab} , with negligible impact on mAP_{img} . The performance of RTN begins to decline faster than LTN with $\pi = WordNet$. This leads to hypothesize that RTN is particularly sensitive to the quality of neighborhoods it is trained on. All models improve monotonically with m .

2) *Joint Models*: We firstly analyze the *naive* case, i.e., $\pi = id$ (Figure 5) and then introduce semantic mapping (Figure 6). The simplest and worst-performing model is LTN+Vecs fed with raw binary vectors; it shows quasi-

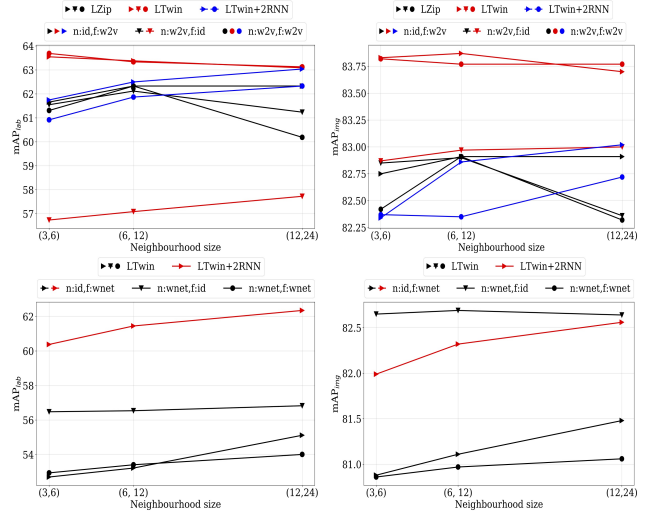


Fig. 6: mAP_{lab} and mAP_{img} for joint models varying the neighborhood size and considering $\pi = w2v$ (1st row) and $\pi = wnet$ (2nd row). Only relevant models and embedding combinations are reported. n refers to the embedding used for neighbors retrieval while f to embedding used to metadata representation.

linear improvement w.r.t. neighborhood. LZIP, which uses a RNN, improves uniformly upon it and achieves very good mAP_{lab} and mAP_{img} from the start but tends to exhibit a mild decrease in performance with neighborhood size, along with LTwin+2RNN. In turn, LTwin achieves good mAP_{img} but comparatively poor mAP_{lab} ; LTwin+RNN achieves roughly comparable performance, but shows linear improvement with m . LZIP, at small (m, M), and LTwin+2RNN are the best-performing models showing that early fusion and RNNs are beneficial to increase network performance, with LTwin comfortably in the middle for mAP_{img} . Unfortunately, LZIP and LTwin+2RNN are also by far the longest to train by an order of magnitude (we just need to consider the breadth of the unrolled graph for non-trivial neighborhood sizes).

The addition of semantic metadata transforms can give a significant boost to performance, in addition to the benefits w.r.t. robustness of the model to vocabulary changes and applicability to a different database than the one used for training. The performance of all architectures is boosted when they are fed transformations computed from word2vec vectors through Eq. 6 instead of plain binary vectors. All models tend to saturate around $(mAP_{lab}, mAP_{img}) = (.63, .83)$. This appears to be the case for LZIP, even without any sort of π . It may be the case that the simpler LTwin can match the performance of the more complex models once provided with word2vec mappings. LTwin (f: word2vec) performs as well as LTwin (n: word2vec, f: word2vec), or even better; the same goes for its LZIP siblings (by a considerably minor margin). We speculate that the ability of the network to learn to take maximal advantage of semantic embeddings overshadows the effect of their use in neighborhood generation and

Method	mAP _{lab}	mAP _{img}	reclab	preclab	recimg	precimg
Tag-only Model + linear SVM [7]	46.67	-	-	-	-	-
Graphical Model (all metadata) [7]	49.00	-	-	-	-	-
CNN + WARP [16]	-	-	35.60	31.65	60.49	48.59
CNN-RNN [21]	-	-	30.40	40.50	61.70	49.90
SR-RNN [22]	-	-	50.17 *	55.65 *	71.35 *	70.57 *
SR-RNN + Vecs [22] †	-	-	58.52 *	63.51 *	77.33 *	76.21 *
SRN [34]	60.00	80.60	41.50 *	70.40 *	58.70 *	81.10 *
MangoNet [32]	62.80	80.80	41.00 *	73.90 *	59.90 *	80.60 *
LTN [2]	52.78 ±0.34	80.34 ±0.07	43.61 ±0.47	46.98 ±1.01	74.72 ±0.16	53.69 ±0.13
LTN + Vecs [2] †	61.88 ±0.36	80.27 ±0.08	57.30 ±0.44	54.74 ±0.63	75.10 ±0.20	53.46 ±0.09
Upper bound	100.00 ±0.00	100.00 ±0.00	65.82 ±0.35	60.68 ±1.32	92.09 ±0.10	66.83 ±0.12
Our baseline: v-only	45.05 ±0.11	76.88 ±0.11	42.31 ±0.59	43.74 ±1.07	71.41 ±0.13	51.36 ±0.13
Our baseline: LTN _{n:id}	53.17 ±0.12	79.82 ±0.16	45.67 ±1.75	47.64 ±2.18	74.29 ±0.13	53.34 ±0.17
Our baseline: LTN + Vecs _{n:id, f:id} †	54.86 ±0.20	81.34 ±0.15	46.56 ±1.39	50.10 ±1.70	75.67 ±0.17	54.37 ±0.14
Our model: RTN _{n:w2v}	55.36 ±0.34	79.77 ±0.27	48.73 ±2.77	51.21 ±2.61	74.35 ±0.29	53.28 ±0.24
Our model: LTwin _{n:w2v, f:w2v} †	63.13 ±0.31	83.77 ±0.06	54.40 ±1.33	51.86 ±1.58	78.06 ±0.05	55.78 ±0.13

TABLE I: Results on NUS-WIDE. We run on 5 splits and report mean and standard deviation. Models that also use metadata are marked with †. In our models n refers to the encoding used to build the neighborhood, while f to the encoding used to represent image metadata. Models such as [22] can decide their own prediction length and are not limited by the parameter k . In these cases (marked with \star) the upper bound does not apply and results are no directly comparable with other approaches.

Arch	n	mAP _{lab}	mAP _{img}
LTN	id	53.17 ±0.12	79.82 ±0.16
LTN	w2v	54.54 ±0.13	80.32 ±0.16
LTN	wnet	53.07 ±0.17	79.95 ±0.24
RTN	id	53.97 ±0.27	79.23 ±0.27
RTN	w2v	55.36 ±0.34	79.77 ±0.27
RTN	wnet	53.76 ±0.33	79.45 ±0.30

TABLE II: Visual Models results for neighborhood size $(m, M) = (12, 24)$. Column n refers to the metadata encoding used to build the neighborhood.

using word2vec vectors in the neighborhood generation process might therefore be unnecessary. LTwin (f : word2vec) emerges as the superior model. As expected, WordNet results in poor performance. Notice also how LTwin (f : WordNet) is particularly sensitive to neighborhood size.

V. CONCLUSION

We have shown that common visual models to classify images, based on metadata to retrieve neighbors, can be improved considering semantic mappings and recurrent neural networks. We have characterized the performance of a variety of visual and joint models and their variability. Our models outperform for several metrics state-of-the-art approaches. We have also shown that semantic mappings can be highly effective in improving performance, besides achieving robustness to changes in metadata vocabulary and quality of neighborhoods.

Acknowledgements: We gratefully acknowledge the support of NVIDIA for their donation of GPUs used in this research. We also acknowledge the UNIPD CAPRI Consortium, for its support and access to computing resources.

REFERENCES

[1] X. Li, T. Uricchio, L. Ballan, M. Bertini, C. Snoek, and A. Del Bimbo, "Socializing the semantic gap: A comparative survey on image tag

Arch	n	f	mAP _{lab}	mAP _{img}
LTN+Vecs	id	id	54.86 ±0.20	81.34 ±0.15
LTN+AllVecs	id	id	56.61 ±0.12	81.28 ±0.21
LZip	id	id	60.64 ±0.14	82.42 ±0.32
LZip	w2v	id	61.24 ±0.51	82.36 ±0.41
LZip	w2v	w2v	60.19 ±0.57	82.32 ±0.15
LZip	id	w2v	62.33 ±0.16	82.91 ±0.18
LTwin	id	id	56.79 ±0.24	82.64 ±0.08
LTwin	id	w2v	63.09 ±0.16	83.70 ±0.14
LTwin	w2v	id	57.73 ±0.17	83.00 ±0.06
LTwin	w2v	w2v	63.13 ±0.31	83.77 ±0.06
LTwin	id	wnet	55.12 ±0.25	81.48 ±0.10
LTwin	wnet	id	56.83 ±0.24	82.64 ±0.10
LTwin	wnet	wnet	54.01 ±0.14	81.06 ±0.10
LTwin+RNN	id	id	58.87 ±0.43	82.95 ±0.08
LTwin+2RNN	id	id	62.00 ±1.44	80.52 ±2.79
LTwin+2RNN	id	w2v	63.04 ±0.22	83.02 ±0.34
LTwin+2RNN	w2v	w2v	62.33 ±0.33	82.72 ±0.37
LTwin+2RNN	id	wnet	62.35 ±0.56	82.56 ±0.26

TABLE III: Joint Models results for neighborhood size $(m, M) = (12, 24)$, and different metadata encodings. Column n refers to the encoding used to build the neighborhood, f to the encoding used as representation: $w2v = \text{word2vec}$, $wnet = \text{wordnet}$, and id refers to raw binary vectors.

assignment, refinement and retrieval," *ACM Computing Surveys*, vol. 49, no. 1, pp. 14:1–14:39, 2016.

- [2] J. Johnson, L. Ballan, and L. Fei-Fei, "Love thy neighbors: Image annotation by exploiting image metadata," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.
- [3] A. Torralba, "Contextual priming for object detection," *International Journal of Computer Vision*, vol. 53, no. 2, pp. 169–191, 2003.
- [4] N. Dvornik, J. Mairal, and C. Schmid, "Modeling visual context is key to augmenting object detection datasets," in *European Conference on Computer Vision (ECCV)*, 2018.
- [5] A. Oliva and A. Torralba, "The role of context in object recognition," *Trends in Cognitive Sciences*, vol. 11, no. 12, pp. 520–527, 2007.
- [6] M. Davis, S. King, N. Good, and R. Sarvas, "From context to content: Leveraging context to infer media metadata," in *ACM International Conference on Multimedia (ACM-MM)*, 2004.
- [7] J. McAuley and J. Leskovec, "Image labeling on a network: Using

- social-network metadata for image classification,” in *European Conference on Computer Vision (ECCV)*, 2012.
- [8] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid, “Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation,” in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2009.
- [9] S. J. Hwang and K. Grauman, “Learning the relative importance of objects from tagged images for retrieval and cross-modal search,” *Int. Journal of Computer Vision*, vol. 100, no. 2, pp. 134–153, 2012.
- [10] Y. Gong, Q. Ke, M. Isard, and S. Lazebnik, “A multi-view embedding space for internet images, tags, and their semantics,” *International Journal of Computer Vision*, vol. 106, no. 2, pp. 210–233, 2014.
- [11] Z. Niu, G. Hua, X. Gao, and Q. Tian, “Semi-supervised relational topic model for weakly annotated image recognition in social media,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [12] J. Hays and A. A. Efros, “IM2GPS: estimating geographic information from a single image,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.
- [13] K. Tang, M. Paluri, L. Fei-Fei, R. Fergus, and L. Bourdev, “Improving image classification with location context,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015.
- [14] G. Wang, D. Hoiem, and D. Forsyth, “Learning image similarity from flickr groups using fast kernel machines,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2177–2188, 2012.
- [15] A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet classification using deep convolutional neural networks,” in *Conference on Neural Information Processing Systems (NeurIPS)*, 2012.
- [16] Y. Gong, Y. Jia, A. Toshev, T. Leung, and S. Ioffe, “Deep convolutional ranking for multilabel image annotation,” in *Int. Conference on Learning Representations (ICLR)*, 2014.
- [17] Y. Verma and C. Jawahar, “Image annotation using metric learning in semantic neighbourhoods,” in *European Conference on Computer Vision (ECCV)*, 2012.
- [18] A. Yu and K. Grauman, “Predicting useful neighborhoods for lazy local learning,” in *Conference on Neural Information Processing Systems (NeurIPS)*, 2014.
- [19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Conf. on Neural Information Proc. Systems (NeurIPS)*, 2013.
- [20] C. Saedi, A. Branco, J. António Rodrigues, and J. Silva, “WordNet embeddings,” in *ACL Wksp on Representation Learning for NLP*, 2018.
- [21] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, “CNN-RNN: A unified framework for multi-label image classification,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [22] F. Liu, T. Xiang, T. M. Hospedales, W. Yang, and C. Sun, “Semantic regularisation for recurrent image annotation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [23] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, “NUS-WIDE: A real-world web image database from national university of singapore,” in *ACM Int. Conference on Image and Video Retrieval (CIVR)*, 2009.
- [24] L.-J. Li and L. Fei-Fei, “OPTIMOL: Automatic online picture collection via incremental model learning,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 147–168, 2010.
- [25] X. Chen and A. Gupta, “Webly supervised learning of convolutional networks,” in *IEEE Int.s Conference on Computer Vision (ICCV)*, 2015.
- [26] C. Rupprecht, A. Kapil, N. Liu, L. Ballan, and F. Tombari, “Learning without prejudice: Avoiding bias in webly-supervised action recognition,” *Comp. Vision and Image Understand.*, vol. 173, pp. 24–32, 2018.
- [27] A. Makadia, V. Pavlovic, and S. Kumar, “A new baseline for image annotation,” in *European Conference on Computer Vision (ECCV)*, 2008.
- [28] V. Lavrenko, R. Manmatha, and J. Jeon, “A model for learning the semantics of pictures,” in *Conference on Neural Information Processing Systems (NeurIPS)*, 2003.
- [29] T. Uricchio, L. Ballan, L. Seidenari, and A. Del Bimbo, “Automatic image annotation via label transfer in the semantic space,” *Pattern Recognition*, vol. 71, pp. 144–157, 2017.
- [30] H. Hu, G.-T. Zhou, Z. Deng, Z. Liao, and G. Mori, “Learning structured inference neural networks with label relations,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [31] M. Guillaumin, J. Verbeek, and C. Schmid, “Multimodal semi-supervised learning for image classification,” in *IEEE Conf. Computer Vision & Pattern Recog. (CVPR)*, 2010.
- [32] J. Zhang, Q. Wu, J. Zhang, C. Shen, and J. Lu, “Mind your neighbours: Image annotation with metadata neighbourhood graph co-attention networks,” in *IEEE Conf. Computer Vision & Pattern Recog. (CVPR)*, 2019.
- [33] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015.
- [34] F. Zhu, H. Li, W. Ouyang, N. Yu, and X. Wang, “Learning spatial regularization with image-level supervisions for multi-label image classification,” in *IEEE Conf. Comp. Vision & Pattern Recog. (CVPR)*, 2017.