Semantic Regularisation for Recurrent Image Annotation

Feng Liu[1,2] Tao Xiang[2] Timothy M. Hospedales[3] Wankou Yang[1] Changyin Sun[1]
[1]Southeast University, China
[2]Queen Mary University of London, UK [3]University of Edinburgh, UK
{liufeng,wkyang,cysun}@seu.edu.cn, {feng.liu,t.xiang}@qmul.ac.uk, t.hospedales@ed.ac.uk

Abstract

The "CNN-RNN" design pattern is increasingly widely applied in a variety of image annotation tasks including multi-label classification and captioning. Existing models use the weakly semantic CNN hidden layer or its transform as the image embedding that provides the interface between the CNN and RNN. This leaves the RNN overstretched with two jobs: predicting the visual concepts and modelling their correlations for generating structured annotation output. Importantly this makes the end-to-end training of the CNN and RNN slow and ineffective due to the difficulty of back propagating gradients through the RNN to train the CNN. We propose a simple modification to the design pattern that makes learning more effective and efficient. Specifically, we propose to use a semantically regularised embedding layer as the interface between the CNN and RNN. Regularising the interface can partially or completely decouple the learning problems, allowing each to be more effectively trained and jointly training much more efficient. Extensive experiments show that state-of-the art performance is achieved on multi-label classification as well as image captioning.

1. Introduction

The classic task of image recognition is beginning to approach a solved problem with the latest Inception-ResNet [26] achieving a top 5 error rate of 3.08% on the ILSVRC15 [24] dataset, surpassing humans. Interest is therefore growing in generating richer descriptions of image properties rather than simple categorisations, including multi-label classification/tagging [13, 15, 14, 31] and image captioning [30, 9, 16, 33, 35, 32].

In multi-label classification the aim is to describe rather than merely recognise an image by annotating all visual concepts that appear in the image. The label space is thus richer than in the single-label recognition case – labels can refer to scene properties, objects, attributes, actions, aesthetics *etc*. Such labels have richer relationships, e.g., a policeman is a person; car and sky co-exist more often than car and sea. Image captioning has a related aim, with the difference of producing a complete natural language sentence description conditioned on the image content, rather than a simple unordered set of labels. For both problems an effective model needs to fulfil two closely-related tasks well: predicting a set of visual concept labels and modelling inter-label correlations. For label-correlation modelling, structured learning strategies are typically employed, which in the case of multi-label classification helps to better distinguish visually ambiguous concepts as well as suppress false predictions (e.g., modelling the car-sky-sea correlation can rectify false prediction of sea in place of sky when a car is present). For image captioning, structured learning is even more critical to generate an ordered list of words that encode a valid as well as relevant sentence.

Recently, the convolutional neural network – recurrent neural network (CNN-RNN) encoder-decoder design pattern has become popular to address the structured label prediction task in both multi-label classification [14, 31] and image captioning [29, 30, 33, 35]. A CNN is used to encode the image into a fixed length vector, which is then fed into an RNN that either decodes it into a list of tags (multi-label) or sequence of words composing a sentence (captioning). With this encoder-decoder architecture, the CNN and RNN can be trained end-to-end, inputting an image and outputting an ordered list of labels. Existing work differs slightly in how the CNN and RNN models are interfaced (see Figs. 1(a)-(c)). However, they share a key characteristic: the image embedding that provides the CNN-RNN interface is the final feature layer of the CNN [14, 22, 31] (e.g. the FC7 layer of Alexnet [18] or the final pooling layer of GoogLeNet [27]) or its linear transform [29, 30].

Using such layers as the input to the RNN has a number of adverse effects on learning an end-to-end recurrent image annotation model. First, since the CNN output feature is not explicitly semantically meaningful, both the label prediction and label correlation/grammar modelling tasks now need to be shouldered by the RNN model alone. This exacerbates the already challenging task of RNN training, since the number of visual concepts/words is often vast (there are more than 12,000 words in the MS COCO training cap-

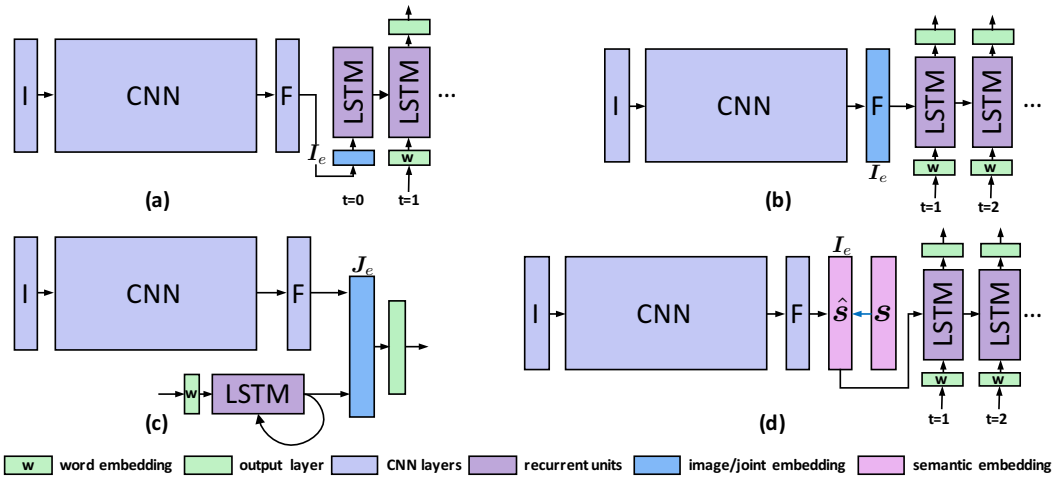

Figure 1. CNN-RNN architectures for image annotation (multi-label classification and captioning). In all models, LSTM is used as the RNN model. (a) CNN encodes an image (I) to a feature representation (F). The image embedding I_e and word representation go through the same word embedding layer before being fed into the LSTM [29]. (b) The image CNN output features F set the LSTM hidden states [14]. (c) The image CNN output feature layer is integrated with the LSTM output via late fusion [22, 31]. (d) The proposed semantically regularised model. The CNN model is regularised by the ground truth semantic concepts $\boldsymbol{s}$, which serve as strong deep supervision to guide the learning of the CNN layers. The CNN prediction layer $\hat{\boldsymbol{s}}$ is used as image embedding which is used to set the LSTM initial states. Best viewed in colour.

tions) and their correlation is rich. Second, a connected CNN-RNN model is effectively rather deep considering the RNN unrolling; existing CNN-RNN models apply supervision only at the final RNN output and propagate the supervision back to the earlier (CNN) layers. This leads to training difficulties in the form of "vanishing" gradients [19]. In addition, joint training of CNN and RNN has to be carried out very carefully to prevent noisy gradients back propagated from the RNN from corrupting the CNN model. As a result, model convergence is often extremely slow [30].

In this paper we propose to change the image embedding layer and introduce semantic regularisation to a CNN-RNN model in order to produce significantly more accurate results and make model training more stable and faster. Specifically, we perform multi-task learning where the auxiliary task (besides tagging/sentence generation) is to regularise the image embedding/interface layer to encode semantically meaningful visual concepts which are directly related to the label prediction task (Fig. 1(d)). This can be understood from several perspectives: (i) As splitting up the system into a model for generating unary potentials (the CNN) by predicting the label individually, and modelling their relations (RNN) for structured prediction. With the unary CNN taking the responsibility of concept prediction, the relational RNN model is better able to focus on learning concept correlations/sentence generation. In the multi-label classification case, where the label space of the semantic regularisation and the RNN output space are the same, this can be seen as analogous to CRF decoding of a joint distribution [36]. (ii) As a deeply supervised network [19], providing auxiliary supervision to the middle of what is effectively a very deep network. Such deep supervision improves accuracy and convergence speed [19, 27]. In our case specifically, it largely eliminates the problem of noisy RNN gradients back-propagating to corrupt the CNN encoder [30]. It thus allows for better and more efficient fine-tuning of the CNN module, as well as fast convergence in end-to-end training of the full CNN-RNN model. (iii) As pursuing an encoder-decoder model with prior bias of preferring semantically meaningful codes [34].

The contributions of this paper are as follows: (1) We propose a novel CNN-RNN image annotation model which differs from the existing models in the selection of the image embedding layer and in the introduction of deeply-supervised semantic regularisation to the embedding layer. (2) Our proposed semantic regularisation enables reliable fine-tuning of the CNN image encoder as well as the fast convergence of end-to-end CNN-RNN training. (3) We demonstrate through extensive experiments that on both multi-label classification and image captioning, we achieve the state-of-the-art performance.

2. Related work

Deep multi-label classification Many earlier studies [15] treat the multi-label classification problem as multiple single label classification problems and ignore the rich correlations in the label space. In order to model label correlation, a structured output model is required. Deng *et al*. [6] propose a hierarchy and exclusion graph (HEX) to model the structure of labels; however, they only fo-

cus on single label classification. Deep structured learning is widely employed in object segmentation. For instance, Zheng *et al*. [36] present an end-to-end structured model that combines the CNN model with a CRF. It allows for fast inference and learning of a deep model with Gaussian edge potentials. This was extended by Chen *et al*. [2] to a deep model which combines MRFs and CNN to model output correlations, and is applied to multi-label classification. Multi-label structure was also effectively modelled by Conditional Graph Lasso [20], but for shallow models.

These CNN-CRF/MRF models work well for image segmentation. However, for multi-label classification, the large label space, seriously imbalanced label distribution, and the need for variable length prediction challenge the application of these models [31]. Recently, the CNN-RNN [14, 31] pattern has been applied to multi-label classification to capture label correlations, as well as address label imbalance and variable length prediction. Since RNN requires sequential input, before training the unordered label set is converted to an ordered list, *e.g*., frequent first [31] or rare first [14]. Small classes can be promoted by using the rare first order. For structured prediction, it is more computationally efficient than CNN-CRF, as it only iterates until the required number of labels are output. Furthermore, it is an end-to-end predictive model as it outputs labels directly, rather than prediction scores, thus eliminating tricky prediction score thresholding heuristics. Our model is related to [14, 31] in that it follows the CNN-RNN design pattern; however, it uses a semantically regularised image embedding layer as the interface layer rather than an unregularised CNN feature layer.

Another line of work is to incorporate side information in multi-label classification, since side information could be complementary to the image data. The side information could be user tags or groups from image metadata [13, 15]. Johnson *et al*. [15] uses a non-parametric approach to find image neighbours according to the metadata, and then aggregates visual information of the image and its neighbours with a deep network to improve classification. In [13] tags, groups, and labels are modelled by different concept layers, which corresponds to different level of abstractions. Messages can be passed top-down and bottom-up by leveraging a bidirectional structured network. Side information can also be exploited in our model, but we show that even using less side information, *e.g*., tags only, our model can outperform those in [13, 15] significantly.

Neural network based image captioning A number of recent captioning studies take a bottom-up approach, where words or phrases are first detected and then composed to sentence with a language model. Fang *et al*. [9] propose a caption model that first detects keywords using a multiple instance learning, and then uses the keywords to generate sentences. A similar model is proposed in [32] with the main difference being that LSTM is used as the language model. Compared with these model, our model is an end-to-end CNN-RNN model which jointly learns the image encoding and language decoding modules.

CNN-RNN based image captioning models have become popular. Vinyals *et al*. [29, 30] follow an encoder-decoder scheme, and feed image features as the initial input to the RNN decoder, so that sentences are generated according to the image. A similar approach is employed in [16]. Our work is related to [29], but we use semantic concepts to regularise the representation of the CNN-RNN interface layer, which leads to significantly improved performance and much easier model training. Recently, visual attention has been incorporated to improve captioning accuracy. Xu *et al*. [33] propose a model capable of sequentially attending to discriminative regions to improve the caption generation. You *et al*. [35] propose to combine visual attributes and image features. An attention mechanism is introduced to reweight attribute predictions and merged with both the input and output of the RNN. Image features are fed at the first step as an external guide. Such attention models could easily be integrated into our model to further improve performance.

Semantic regularisation in deep encoder-decoders The idea of introducing semantic regularisation to an encoder-decoder model has been exploited in the context of image synthesis. Yan *et al*. [34] extend the variational autoencoder [17] by introducing attribute induced semantic regularisation to the middle embedding layer. A similar model based on generative adversarial networks is also proposed [23]. Despite the similar strategy to ours, the objective is very different: we use the encoder-decoder architecture to align the text and image modalities and middle-layer supervision is employed to achieve more effective and efficient training of both the encoder and decoder.

3. Methodology

We first give an overview of existing CNN-RNN models before introducing our semantically regularised CNN-RNN. Its application to multi-label classification and image captioning are detailed in Sec. 4 and Sec. 5 respectively.

3.1. CNN-RNN

A CNN-RNN model is composed of two parts: a visual encoder perceives the visual content of an image and encodes it to an image embedding; and a decoder takes the embedding as input and generates sequences of labels (words).

Given an image I, a visual encoder will encode it to a fixed length vector $\boldsymbol{I}_e \in \mathbb{R}^{d\times 1}$ called image embedding:

$$\boldsymbol{I}_e = f_{enc}(I), \tag{1}$$

where f_{enc} is the encoder, which could be a pretrained CNN optionally with some additional transformation layers. So

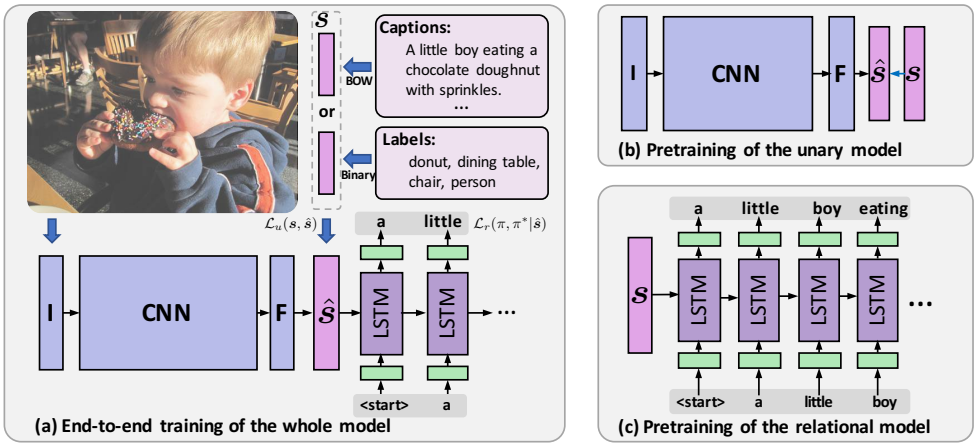

Figure 2. The full pipeline of the proposed semantically regularised annotation model. The ground truth semantic concepts serve as strong supervision in the middle to regularise the training of the unary model (a). Due to the use of semantic concepts as the interface between CNN and RNN, the unary model and relational models can be pretrained in parallel, as shown in (b), (c).

$\boldsymbol{I}_e$ could either be a feature layer [14, 22, 31], *e.g.*, FC7 layer of VGG16 [25], or its linear transform [29, 30]. In this paper, we enforce it to be a semantic representation to better interact with the RNN.

The RNN decoder will then take $\boldsymbol{I}_e$ as a condition, and generate a predictive path $\boldsymbol{\pi} = (a_1, a_2, ..., a_{n_s})$, where for multi-label classification, a_i is semantic label, and n_s is the number of labels predicted for image I; while for image captioning a_i is the word token, and n_s is the length of the sentence. The path is an ordered sequence, so in multi-label classification, a priority of the labels has to be defined to convert labels to a sequence. We take a rare first order so as to give rare classes more importance during the prediction, therefore countering the label imbalance problem.

Many different CNNs have been considered for the encoder, but for the RNN decoder, the long short-term memory (LSTM) model [12] has been chosen by almost all existing models. This is because it controls message passing between times steps with gates in order to alleviate the vanishing/exploding gradient problem which plagued the training of prior RNN models. The model has two types of states: cell state $\boldsymbol{c}$ and hidden state $\boldsymbol{h}$. Following [11], a forward pass at time t with input $\boldsymbol{x}_t$ is computed as follows.

$$
\begin{aligned}
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_{i,h} \cdot \boldsymbol{h}_{t-1} + \boldsymbol{W}_{i,c} \cdot \boldsymbol{c}_{t-1} + \boldsymbol{W}_{i,x} \cdot \boldsymbol{x}_t + \boldsymbol{b}_i) \\
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_{f,h} \cdot \boldsymbol{h}_{t-1} + \boldsymbol{W}_{f,c} \cdot \boldsymbol{c}_{t-1} + \boldsymbol{W}_{f,x} \cdot \boldsymbol{x}_t + \boldsymbol{b}_f) \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_{o,h} \cdot \boldsymbol{h}_{t-1} + \boldsymbol{W}_{o,c} \cdot \boldsymbol{c}_{t-1} + \boldsymbol{W}_{o,x} \cdot \boldsymbol{x}_t + \boldsymbol{b}_o) \\
\boldsymbol{g}_t &= \delta(\boldsymbol{W}_{g,h} \cdot \boldsymbol{h}_{t-1} + \boldsymbol{W}_{g,c} \cdot \boldsymbol{c}_{t-1} + \boldsymbol{W}_{g,x} \cdot \boldsymbol{x}_t + \boldsymbol{b}_g) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \delta(\boldsymbol{c}_t)
\end{aligned}
\tag{2}
$$

where $\boldsymbol{c}_t$ and $\boldsymbol{h}_t$ are the model's cell and hidden states, $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$ are the activation of input gate, forget gate, and output gate respectively; $\boldsymbol{W}_{\cdot,h}$, $\boldsymbol{W}_{\cdot,c}$ are the recurrent weights, and $\boldsymbol{W}_{\cdot,x}$ is the input weight, and $\boldsymbol{b}_\cdot$ are the biases. $\sigma(\cdot)$ is the sigmoid function, and δ is the output activation function.

At time step t, the model uses its last prediction a_{t-1} as input, and computes a distribution over possible outputs:

$$
\begin{aligned}
\boldsymbol{x}_t &= \boldsymbol{E} \cdot \boldsymbol{a}_{t-1}, \\
\boldsymbol{h}_t &= LSTM(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}, \boldsymbol{c}_{t-1}), \\
\boldsymbol{y}_t &= softmax(\boldsymbol{W} \cdot \boldsymbol{h}_t + \boldsymbol{b}),
\end{aligned}
\tag{3}
$$

where $\boldsymbol{E}$ is the word embedding matrix, $\boldsymbol{h}_{t-1}$ is the hidden state of the recurrent units at $t-1$, $\boldsymbol{W}$, $\boldsymbol{b}$ are the weight and bias of the output layer, $\boldsymbol{a}_{t-1}$ is the one-hot coding of last prediction a_{t-1}, and $LSTM(\cdot)$ is a forward step of the unit. The output $\boldsymbol{y}_t$ defines a distribution over possible actions, from which the next action a_{t+1} is sampled.

To generate image-conditioned sequences, the decoder has to take advantage of the image embedding $\boldsymbol{I}_e$, and existing models achieve this in multiple ways. Vinyals *et al.* [29] (Fig. 1(a)) propose to feed $\boldsymbol{I}_e$ as step zero input to the LSTM model, that is, $(\boldsymbol{h}_0, \boldsymbol{c}_0) = LSTM(\boldsymbol{I}_e, \boldsymbol{0}, \boldsymbol{0})$, where $\boldsymbol{0}$ is a zero vector. In this case the weights of the word embedding are shared with image embedding, which is a questionable assumption, as the two embeddings have very different meanings and their dimensions have not been aligned. Instead of treating $\boldsymbol{I}_e$ as an LSTM input, Wang *et al.* [31] and Mao *et al.* [22] combine word embedding and image features via output fusion (Fig. 1(c)). In contrast, Jin *et al.* [14] use the image embedding to initialise the LSTM (Fig. 1(b)) by setting hidden state $\boldsymbol{h}_0 = \boldsymbol{W}_i \cdot \boldsymbol{I}_e + \boldsymbol{b}_i$, where $\boldsymbol{W}_i$, $\boldsymbol{b}_i$ are image input weights and biases.

Despite these differences, existing CNN-RNN models have a key common characteristic: The image embedding

$\boldsymbol{I}_e$ that acts as the interface between the CNN and RNN models is taken to be a layer of weak and implicit semantics, *e.g.*, CNN feature layer, or its transform. This means that the RNN has to simultaneously learn to predict semantic concepts from the provided features, as well as model the correlation of those concepts. Learning to predict the concepts is harder for the RNN because gradients are back propagated from relatively 'far' supervision away (the RNN outputs at future time steps). Moreover fine-tuning the CNN becomes tricky because noisy gradients propagated from the RNN can easily degrade rather than improve performance [30].

3.2. Semantically regularised CNN-RNN

To reduce the burden on the RNN, we propose a divide-and-conquer strategy to separate two tasks: semantic concept learning and relational modelling. Specifically, semantic concept learning is now performed by the unary CNN model which takes as input images (and associated side information if any), and produces a probabilistic estimate of the semantic concepts. Relational modelling is handled by the RNN model which takes in the concept probability estimates and models their correlations to generate label/word sequences. Concretely, instead of using a CNN feature layer as embedding $\boldsymbol{I}_e$, we use the CNN label prediction layer, *e.g.*, concept prediction layer of an Inception net [28]. Since the chosen embedding is trained under direct supervision of ground-truth labels/visual concepts, it has clear semantic meaning: Each unit corresponds to a semantic concept.

As shown in Fig. 2, in our Semantically regularised CNN-RNN (S-CNN-RNN), the CNN part takes an image I as input, and predicts the likelihood of the semantic concepts $\hat{\boldsymbol{s}} \in \mathbb{R}^{k\times 1}$ where k is the number of semantic concepts[1]. The RNN model takes $\hat{\boldsymbol{s}}$ as input, and generates sequences π. The key implication is that supervision can now be added at *both* the RNN output layer and the embedding layer $\hat{\boldsymbol{s}}$. This results in two losses: a loss for concept prediction $\mathcal{L}_u(\boldsymbol{s}, \hat{\boldsymbol{s}})$ and a loss for relational modelling $\mathcal{L}_r(\pi, \pi^*|\hat{\boldsymbol{s}})$. Formally, we have

$$\begin{aligned} \mathcal{L}_u(\boldsymbol{s}, \hat{\boldsymbol{s}}) &= \sum_i \ell_u(\boldsymbol{s}_i, \hat{\boldsymbol{s}}_i) \\ \mathcal{L}_r(\pi, \pi^*|\hat{\boldsymbol{s}}) &= \sum_i \ell_r(\pi_i, \pi_i^*|\hat{\boldsymbol{s}}_i) \\ \mathcal{L} &= \mathcal{L}_u(\boldsymbol{s}, \hat{\boldsymbol{s}}) + \mathcal{L}_r(\pi, \pi^*|\hat{\boldsymbol{s}}), \end{aligned} \quad (4)$$

where $\boldsymbol{s}_i$ is the ground truth concept labels for the i-th training image and $\hat{\boldsymbol{s}}_i$ is the corresponding prediction; For the RNN loss $\mathcal{L}_r(\pi, \pi^*|\tilde{\boldsymbol{s}})$, π_i^* is the ground truth path; π_i is the predicted path, which is a sequence of word tokens or list of tags. The specific form of the losses will be discussed next.

[1] k is the size of label space in multi-label classification. For image captioning, k is the number of visual concepts, which is typically smaller than the vocabulary size as not all words are visual.

3.3. Training and inference

The introduction of semantic regularisation in the middle of CNN-RNN allows for more effective and efficient model training. It facilitates a two-staged training strategy illustrated in Fig. 2. In the first stage, we pretrain the CNN model and RNN model in parallel and in the second stage, they are fine-tuned together.

CNN For pretraining of the CNN model (Fig. 2(b)), the ground truth semantic concepts $\boldsymbol{s}_i$ are used as the learning target in a standard cross entropy loss for k visual concepts:

$$\ell_u(\boldsymbol{s_i}, \hat{\boldsymbol{s}}_i) = \sum_j^k s_{ij} \cdot \log(\hat{s}_{ij}) + (1 - s_{ij}) \cdot \log(1 - \hat{s}_{ij}), \quad (5)$$

LSTM For the LSTM pretraining (Fig. 2(c)), the concept input $\hat{\boldsymbol{s}}_i$ is first connected to a fully connected (FC) layer before being used to set the initial hidden state of the LSTM[2]. The LSTM model learns to maximise the likelihood of generating the target sequences conditioned on the semantic input, and the loss $\mathcal{L}_r(\pi, \pi^*|\hat{\boldsymbol{s}})$ is simply the sum of the negative log likelihood over all time steps. By feeding $\boldsymbol{s}$, rather than $\hat{\boldsymbol{s}}$ the LSTM can be pre-trained independently of the CNN.

Joint CNN-LSTM After the CNN and RNN models are pretrained, the whole model can be jointly trained by simultaneously optimising the deeply supervised joint loss $\mathcal{L}$. For inference, we condition on the image by setting the initial state, then feed a start signal and recurrently sample model predictions of the previous step as input until an end signal is generated. For multi-label classification, we just greedily take the maximum model output, whilst beam search with a width of three is employed for image captioning [30].

4. Application to Multi-label Classification

4.1. Formulation

To apply our S-CNN-RNN to multi-label classification, we first rank the training labels according to their frequency in the training set and generate a ordered label list with the rare labels first. We also explore the use of side information [15, 13]: exploiting the noisy user-provided tags available with each image. In this case the model in Fig. 2 is slightly modified. Specifically, we pretrain a multiple layer perception (MLP) (single 256 neuron hidden layer and ReLU activation) to predict the true tags given the noisy metadata. Then we combine the image model with the pretrained tag model by summing their predictions as the final embedding $\hat{\boldsymbol{s}}$), and train them together with a cross entropy loss [37].

[2] This is to allow for the flexibility of using arbitrary LSTM unit size.

4.2. Datasets and settings

Datasets Two widely used large-scale benchmark datasets are selected to evaluate our model. **NUS-WIDE** [5] dataset contains 269,648 images. Originally coming from Flickr, there are 5,018 unique user tags released along with the images. Of them, 81 tags are manually selected and refined as the ground truth [5], covering different aspects including object classes, scenes, and attributes. The ground truth labels are highly imbalanced: the most frequent tag, *sky* appears 74,190 times while the rarest one *map* appears 60 times. In addition, the user-provided tags are extremely noisy and sparse – 8.73 noisy tags per image on average. Following [15, 13], we consider two settings: multi-label classification with only imagery data and with both images and noisy tags as side information. The most popular 1,000 noisy user tags are kept and we remove the images without any ground-truth tags. As in many Flicker based studies, the numbers of images used by different works vary as they download the images at different times. For fair comparison, we use the same train/test split ratio as [15, 13]; as a result, 15,000 images are used for training and 59,347 for testing. **Microsoft COCO** [21] is popular for tasks such as object detection, segmentation and image captioning. Following [31], we also use it for multi-label classification by treating the 80 object classes as labels. Since there are normally many types of objects in each image, it is naturally a multi-label classification problem. Because the label space contains objects only and some objects are rather small, it is perhaps more suitable than NUS-WIDE for evaluating a structured prediction model, as modelling label correlation becomes more important to detect visually similar and small objects. We also download the original user tags from Flickr via the provided URLs, and the most frequent 1,000 tags are used as side information. We keep the original train/validation split [21] for training and evaluation.

Implementation details For fair comparisons with previous work, in our S-CNN-RNN model, we use the caffe reference net [8] as our unary CNN subnet on the NUS-WIDE dataset [5], and VGG16 on MS COCO. Both models are pretrained on the ILSVRC12 dataset [24]. For pretraining the CNN subnet, the learning rate is set to 1e-4 for NUS-WIDE and 1e-3 for MS COCO. For the RNN subnet, we use 512 LSTM cells and a 256 dimensional word embedding. The output vocabulary size is set to 82 for NUS-WIDE and 81 for MS COCO, including all labels and an END token. We use the `BasicLSTMCell` in TensorFlow as LSTM cells and employ ReLU as activation function. The relational model is trained using a RMS Prop optimiser with a learning rate of 1e-4.

Evaluation metrics As in [14, 31], both per-class and per-image metrics including mean precision and mean recall are used. For each class/image, the precision is defined as: $p(\hat{y}, y) = |y \cap \hat{y}|/|\hat{y}|$; and recall is defined as: $r(\hat{y}, y) = |y \cap \hat{y}|/|y|$, where y and $\hat{y}$ are the set of ground truth labels and predicted labels, and $|\cdot|$ is the cardinality of a set. The overall precision (O-P)/recall (O-R) is computed by taking the average precision/recall over all samples, while the per class precision (C-P)/recall (C-R) is averaged over all classes. F1 score is also computed by computing the harmonic mean of precision and recall. As in existing CNN-RNN models [14, 31], we let the model to decide its own prediction length [14, 31], whilst for other compared fixed-length predictive models [13, 15, 31], we use the top 3 ranked predictions.

4.3. Experimental results

Competitors We compare with the following models. In all compared models, the same CNN and RNN modules are used. **CNN+Logistic:** This model treats each label independently by fitting a logistic regression classifier for each label. The results are reported in [13]. **CNN+Softmax:** A CNN model that uses softmax as classifier, and the cross entropy between prediction and ground truth is used as the loss function. The results reported in [10] for NUS-WIDE and [31] for MS COCO are used. **CNN+WARP:** Same CNN model as above, but uses a weighted approximate ranking loss function for training to promote the prec@K metric. We use the results reported in [10] for NUS-WIDE and [31] for MS COCO. **CNN-RNN:** A CNN-RNN model which uses output fusion (Fig. 1(c)) to merge CNN output features and RNN outputs [31]. **RIA:** In this CNN-RNN model [14], the CNN output features are used to set the LSTM hidden state (Fig. 1(b)). Note that only smaller datasets were used in [14] and no code is available; we thus use our own carefully trained implementation in the experiments. **Tag-Neighbour:** It uses a non-parametric approach to find image neighbours according to metadata, and then aggregates image features for classification. Tag neighbour with 5K tags gives the best performance [15]. It uses more side information than ours and is also transductive requiring access to the whole test set at once. **SINN:** It [13] uses different concept layers of tags, groups, and labels to model the semantic correlation between concepts of different abstraction levels. A bidirectional RNN-like algorithm is adopted to integrate information for prediction. 1K noisy tags and 698 query words are used as side information, which is more than what our model uses. **Variants of our model:** Our S-CNN-RNN with and without the side information are called Ours and Ours+Tag1K respectively. Since the results reported by SINN [13] and TagNeighbour [15] were based on ImageNet-pretrained CNN models, for direct comparison we train a variant of our model that fixes the weights of the CNN subnet without finetuning (Ours+Tag1K Fix).

Results on NUS-WIDE We make the following observations from the results shown in Table 1. (1) The proposed S-CNN-RNN performs consistently better than all alternatives

Algorithms	C-R	C-P	C-F1	O-R	O-P	O-F1
CNN+logistic [13]	45.03	45.60	45.31	70.77	51.32	59.50
CNN+Softmax [10]	31.22	31.68	31.45	59.52	47.82	53.03
CNN+WARP [10]	35.60	31.65	33.51	60.49	48.59	53.89
CNN-RNN [31]	30.40	40.50	34.70	61.70	49.90	55.20
RIA [14]	43.62	52.92	47.82	66.75	68.98	67.85
TagNeighboor† [15]	57.30	54.74	55.99	75.10	53.46	62.46
SINN† [13]	60.63	58.30	59.44	**79.12**	57.05	66.30
Ours	50.17	55.65	52.77	71.35	70.57	70.96
Ours+Tag1K Fix†	58.52	63.51	60.91	77.33	76.21	76.77
Ours+Tag1K†	**61.73**	**71.73**	**66.36**	76.88	**77.41**	**77.15**

Table 1. Multi-label classification results on NUS-WIDE. Results that use side information are marked with superscript †.

in terms of the F1 score, both with (Ours+Tag1K) and without side information (Ours). (2) Looking at the precision and recall metrics, our model is more impressive on precision than recall. This is expected because compared to the non-CNN-RNN based models that predict a fixed number of 3 labels, a CNN-RNN model tends to makes less predictions for this dataset with on average 2.4 ground truth tags per image. (3) The gaps between Ours and CNN-RNN [31] and RIA [14] show clearly the importance of adding semantic regularisation to the CNN embedding layer. (4) Comparing Ours+Tag1K Fix with TagNeighboor [15] and SINN [13], we can see that significant improvements are obtained even with less side information. This is due to the ability of the RNN decoder in our CNN-RNN model to model high-order label correlations. (5) Our full model (Ours+Tag1K) further improves over Ours+Tag1K Fix on both per class and per image metric. This shows the importance of having an end-to-end CNN-RNN that can be trained effectively with the introduced deeply supervised semantic regularisation. Qualitative results can be found in the supplementary material.

Results on MS COCO Similar conclusions can be drawn from the results in Table 2. Comparing with the results on NUS-WIDE, it is noted that the performance gain obtained by using the 1K noisy tags as side information is smaller. This is because that the number of user-provided tags on COCO is smaller (2.93 vs. 6.10 per image with 1K unique tags).

Algorithms	C-R	C-P	C-F1	O-R	O-P	O-F1
CNN+logistic [31]	58.60	59.30	58.90	65.00	61.70	63.30
CNN+Softmax [31]	59.00	57.00	58.00	60.20	62.10	61.10
CNN+WARP [31]	59.30	52.50	55.70	59.80	61.40	60.70
CNN-RNN [31]	55.60	66.00	60.40	66.40	69.20	67.80
RIA [14]	54.07	64.32	58.75	64.57	74.20	69.05
Ours	59.83	67.40	63.39	68.73	76.63	72.47
Ours+Tag1K†	**63.13**	**71.38**	**67.00**	**73.05**	**77.41**	**75.16**

Table 2. Multi-label classification results on Microsoft COCO.

5. Application to Image Captioning

5.1. Datasets and settings

Datasets and metrics We use the popular Microsoft COCO dataset [21] for evaluation. The dataset contains 82,783 training images and 40,504 validation images. Each image is manually annotated with 5 captions. The comparison against the state-of-the-art is conducted using the actual MS COCO test set comprising 40,775 images. Note that the annotation of the test set is not publicly available, so the results are obtained from the COCO evaluation server. For an ablation study, we also follow the setting of [29, 30] by a held-out set of 4,051 images from the validation set as the test set. The widely used BLEU, CIDEr, METEOR, and ROUGE scores are employed to measure the quality of generated captions. For the ablation study, they are computed using the `coco-evaluation` code [3].

Implementation details For our S-CNN-RNN, we use Inception v3 [28] as the CNN subnet, and an LSTM network is used as RNN subnet. The number of LSTM cells is 512, equalling to the dimension of the word embedding. The output vocabulary size for sentence generation is 12,000. Note that all these are exactly the same as the NIC v2 [30] model ensuring a fair comparison. For semantic regularisation by deep supervision of image embedding layer, we need to extract a set of semantic concepts/training labels from the vocubulary. To this end, we follow [9] and simply use the 1,000 most frequent words in the captions, which cover 92% of word occurrences. The ground truth labels for a training image is defined as the words that appear at least once in the 5 captions. For the CNN pretraining, we initially just learn the prediction layer, and then tune all the parameters for 30,000 iterations with a batch size of 32 and learning rate of 1e-4. In parallel, the RNN model is pretrained for 1,000,000 iterations with the ground truth semantic labels as image embedding. After both models are pretrained, the full model is fine-tuned for 500,000 iterations.

5.2. Experimental results

Competitors Five state-of-the-art models are selected for comparison: **MSRCap**: The Microsoft Captivator [7] combines the bottom-up based word generation model [9] with a gated recurrent neural network [4] (GRNN) for image captioning. **mRNN**: The multimodal recurrent neural network [22] uses a multimodal layer to combine the CNN and RNN. **NICv2**: The NICv2 [30] is an improved version of the Neural Image Caption generator [29]. It uses a better image encoder Inception V3. In addition, scheduled sampling [1] and an ensemble of 15 models are used; both improved the accuracy of captioning. Neither is used in our model. **V2L**: The V2L model [32] use a CNN based attribute detector to firstly generate 256 attributes, and then feed as initial input to a LSTM model to generate captions. **ATT**: The se-

Metric	B-1		B-2		B-3		B-4		METEOR		ROUGE		CIDEr	
	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40
MSRCap [7]	0.715_{20}	0.907_{8}	0.543_{19}	0.819_{9}	0.407_{19}	0.710_{10}	0.308_{16}	0.601_{10}	0.248_{16}	0.339_{11}	0.526_{19}	0.680_{14}	0.931_{15}	0.937_{16}
mRNN [22]	0.716_{18}	0.890_{20}	0.545_{18}	0.798_{20}	0.404_{20}	0.687_{20}	0.299_{21}	0.575_{20}	0.242_{26}	0.325_{25}	0.521_{23}	0.666_{24}	0.917_{18}	0.935_{17}
V2L [32]	0.725_{10}	0.892_{18}	0.556_{11}	0.803_{17}	0.414_{14}	0.694_{17}	0.306_{18}	0.582_{18}	0.246_{19}	0.329_{21}	0.528_{16}	0.672_{18}	0.911_{20}	0.924_{20}
NICv2 [30]	0.713_{21}	0.895_{17}	0.542_{21}	0.802_{18}	0.407_{18}	0.694_{18}	0.309_{15}	0.587_{16}	0.254_{8}	0.346_{6}	0.530_{15}	0.682_{11}	0.943_{12}	0.946_{14}
ATT [35]	0.731_{9}	0.900_{14}	0.565_{9}	0.815_{11}	0.424_{8}	0.709_{11}	0.316_{9}	0.599_{11}	0.250_{13}	0.335_{17}	0.535_{8}	0.682_{12}	0.943_{11}	0.958_{11}
Ours	$\mathbf{0.743}_{5}$	$\mathbf{0.917}_{4}$	$\mathbf{0.578}_{5}$	$\mathbf{0.840}_{4}$	$\mathbf{0.434}_{6}$	$\mathbf{0.735}_{5}$	$\mathbf{0.323}_{6}$	$\mathbf{0.621}_{5}$	$\mathbf{0.255}_{7}$	$\mathbf{0.343}_{7}$	$\mathbf{0.540}_{6}$	$\mathbf{0.691}_{5}$	$\mathbf{0.986}_{6}$	$\mathbf{1.002}_{5}$

Table 3. Results from the official MS-COCO testing server (https://www.codalab.org/competitions/3221#results). The subscript indicates the ranking by the submission date (15th/Nov/2016) w.r.t. each metric.

mantic attention model [35] uses both image features and visual attributes, and introduces an attention mechanism to reweight the attribute context to improve captioning accuracy. All five models use a CNN and a RNN, but only NICv2 does end-to-end training. In contrast, ATT does attention model and RNN joint training, and uses a 5-model ensemble. There is no joint training for the other three.

Results We submit our results to the official evaluation server to compare with the five baselines which also appear in the official ranking. The evaluation is done with both 5 and 40 reference captions (C5 and C40). It can be seen from Table 3 that our model beats all five competitors on all 14 metrics, often by a significant margin. Among the 39 submitted models, our model is ranked the 5th and we could not find references for the four higher ranked models. Note that our performance across all metrics is very consistent. In contrast, the 5 competitors often do well on some metrics but very badly on others. It is worth pointing out that our result is obtained without a model ensemble, a practice commonly used in this type of benchmarking exercise (e.g., both NICv2 and ATT use ensembles). In addition, no auxiliary captioning data is used for training. This result thus represents the state-of-the-art. For qualitative results please see the supplementary material.

Ablation study We compare our full model with two stripped-down versions. **NIC-F**: removing the semantic regularisation and use the CNN output feature layer as the inference $\boldsymbol{I}_e$ to RNN. This gives us the standard NIC model [29] with the same Inception v3 as CNN subnet. The model is finetuned end-to-end on COCO. **NIC-deeply**: this model is closer to ours – it uses the same deeply supervised semantic regularisation as our model, but the penultimate feature layer is taken as the embedding, rather than the prediction layer $\hat{\boldsymbol{s}}$. As a result, the CNN feature representation benefits from the deep supervision (rather than distal supervision via the RNN), but the specific embedding used as the RNN interface is not directly semantically meaningful. The results on the validation set split are shown in Table 4. It can be seen that: (1) Semantic regularisation is critical, e.g., it brings about 7% on CIDEr comparing NIC-F and our full model. (2) The deep supervision is the most crucial contributor to the good performance of our model. Even when the embedding layer is not semantically explicit as in NIC-deeply, the benefit is evident. The smaller gap between NIC-deeply and Ours is due to the use of the semantically explicit prediction layer as the embedding at the CNN-RNN interface.

Metric	CIDEr	METEOR	ROUGE	B-4
NIC-F	0.932	0.247	0.524	0.297
NIC-deeply	1.006	0.258	0.543	0.323
Ours	**1.054**	**0.260**	**0.550**	**0.340**

Table 4. Ablation study results on the COCO validation set split.

Computational cost Thanks to the semantic regularisation, the proposed model can be trained very efficiently. The total training takes two days on a single Nvidia Titan X GPU. In contrast training one of NIC's 15-model ensemble members takes more than 20 days on the same GPU. In particular, the deep supervision allows the model to converge very fast. For example, pretraining our Inception v3 [28] CNN only needs 30,000 iterations with a batch size of 32. The pretraining of the RNN model is also fast since its inputs are ground truth labels. After the pretraining, the full model fine-tuning converges much faster than NICv2.

6. Conclusion

We proposed a semantically regularised CNN-RNN model for image annotation. The semantic regularisation makes the CNN-RNN interface semantically meaningful, distributes the label prediction and correlation tasks between the CNN and RNN models, and importantly the deep supervision makes training the full model more stable and efficient. Extensive evaluations on NUS-WIDE and MS-COCO demonstrate the efficacy of the proposed model on both multi-label classification and image captioning.

Acknowledgements: This project received support from Natural Science Foundation of China (NSFC) grant #61473086, the Scientific Research Foundation of Graduate School of Southeast University grant #YBJJ1520, and the China Scholarship Council (CSC). We gratefully acknowledge the support of NVIDIA Corporation for the donation of the GPUs used for this research.

References

[1] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*, 2015. 7

[2] L.-C. Chen, A. G. Schwing, A. L. Yuille, and R. Urtasun. Learning deep structured models. In *ICML*, 2015. 3

[3] X. Chen, H. Fang, T. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. Microsoft COCO captions: Data collection and evaluation server. *CoRR*, abs/1504.00325, 2015. 7

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 7

[5] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. NUS-WIDE: A real-world web image database from national university of singapore. In *CIVR*, 2009. 6

[6] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam. Large-scale object classification using label relation graphs. In *ECCV*, 2014. 2

[7] J. Devlin, H. Cheng, H. Fang, S. Gupta, L. Deng, X. He, G. Zweig, and M. Mitchell. Language models for image captioning: The quirks and what works. In *ACL*, 2015. 7, 8

[8] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014. 6

[9] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, et al. From captions to visual concepts and back. In *CVPR*, 2015. 1, 3, 7

[10] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013. 6, 7

[11] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013. 4

[12] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997. 4

[13] H. Hu, G.-T. Zhou, Z. Deng, Z. Liao, and G. Mori. Learning structured inference neural networks with label relations. In *CVPR*, 2016. 1, 3, 5, 6, 7

[14] J. Jin and H. Nakayama. Annotation order matters: Recurrent image annotator for arbitrary length image tagging. In *ICPR*, 2016. 1, 2, 3, 4, 6, 7

[15] J. Johnson, L. Ballan, and L. Fei-Fei. Love thy neighbors: Image annotation by exploiting image metadata. In *ICCV*, 2015. 1, 2, 3, 5, 6, 7

[16] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015. 1, 3

[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013. 3

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[19] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, 2015. 2

[20] Q. Li, M. Qiao, W. Bian, and D. Tao. Conditional graphical lasso for multi-label image classification. In *CVPR*, 2016. 3

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6, 7

[22] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). In *ICLR*, 2015. 1, 2, 4, 7, 8

[23] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text-to-image synthesis. In *ICML*, 2016. 3

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 1, 6

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 4

[26] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016. 1

[27] 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. 1, 2

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 5, 7, 8

[29] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 1, 2, 3, 4, 7, 8

[30] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: Lessons learned from the 2015 MSCOCO image captioning challenge. *TPAMI*, 2016. 1, 2, 3, 4, 5, 7, 8

[31] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu. CNN-RNN: A unified framework for multi-label image classification. In *CVPR*, 2016. 1, 2, 3, 4, 6, 7

[32] Q. Wu, C. Shen, L. Liu, A. Dick, and A. van den Hengel. What value do explicit high level concepts have in vision to language problems? In *CVPR*, 2016. 1, 3, 7, 8

[33] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015. 1, 3

[34] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2image: Conditional image generation from visual attributes. In *ECCV*, 2016. 2, 3

[35] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *CVPR*, 2016. 1, 3, 8

[36] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *CVPR*, 2015. 2, 3

[37] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam, and R. Fergus. Simple baseline for visual question answering. *arXiv preprint arXiv:1512.02167*, 2015. 5