Expert Systems With Applications 182 (2021) 115195

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Condition-CNN: A hierarchical multi-label fashion image classification model

Brendan Kolisnik, Isaac Hogan *, Farhana Zulkernine

School of Computing, Queen's University, Kingston, ON K7L2N8, Canada

ARTICLE INFO

Keywords:
Condition-CNN
Branching convolutional neural networks
Image classification
Convolutional neural networks
Hierarchical image classification
Teacher Forcing

ABSTRACT

Current state of the art image classifiers predict a single class label of an image. However, in many industry settings such as online shopping, images belong to a class hierarchy where the first level represents the coarse grained or the most abstract class with subsequent levels representing the more specific classes. We propose a novel hierarchical image classification model, Condition-CNN, which addresses some of the shortcomings of the branching convolutional neural network in terms of training time and fine-grained accuracy. It applies the Teacher Forcing training algorithm, where the actual class labels of the higher level classes rather than the predicted labels are used to train the lower level branches. The technique also prevents error propagation, and thereby, reduces the training time. Besides learning the image features for each level of classes, Condition-CNN also learns the relationship between different levels of classes as conditional probabilities, which is used to estimate class predictions during scoring. By feeding the estimated higher-level class predictions as priors to the lower-level class prediction, Condition-CNN achieves a superior prediction accuracy while requiring fewer trainable parameters compared to the baseline CNN models. The validation results of Condition-CNN using the Kaggle Fashion Product Images data set demonstrate a prediction accuracy of 99.8%, 98.1%, and 91.0% for Level 1, 2 and 3 classes respectively, which are greater than that of B-CNN and other baseline CNN models. Moreover, Condition-CNN used only 77.58% of the total number of trainable parameters as that of B-CNN.

1. Introduction

According to the Grand View Research report of March 2020 (Grand View Research, 2020), the market value of image recognition is predicted to become $109.4 billion by 2027. The market is anticipated to expand at a compound annual growth rate of 18.8% from 2020 to 2027 due to the increasing success of image recognition applications in a variety of domains including medical, financial, security surveillance, and the online marketplace. Most of the image classification approaches today apply supervised machine learning where models are trained to predict a class of a given image from a single level of classes. Among the different computer vision techniques, traditional computational models apply complex handcrafted computing algorithms to extract the desired set of features from specific parts of an image (Bay et al., 2006; Wang et al., 2018). Over the last decade with the advancements in the GPU and computing power, deep learning algorithms, specifically deep convolutional neural network (CNN) models, have shown great success (Krizhevsky et al., 2012). Given a large number of images, these models learn to automatically extract spatial features from the images using a series of convolutional and max pooling layers and optimize the parameters. The extracted features are fed to one or more classification layers which predict the image class at the final output layer. Supervised learning algorithms use labeled data to learn data features and optimize model parameters through multiple training iterations until an acceptable accuracy is achieved.

Supervised learning has been widely used in state-of-the-art computer vision algorithms for image classification (Bay et al., 2006), object localization, detection (Hara et al., 2016), recognition (Li et al., 2019; Qiao & Zulkernine, 2020b; Wojaczek et al., 2019), and object or image segmentation (Chen et al., 2015). Challenges in these problems include handling noisy images or videos (Qiao & Zulkernine, 2020a, 2020b; Wojaczek et al., 2019), recognizing partially occluded objects (Gasmallah & Zulkernine, 2018), processing spatio-temporal data (Gasmallah & Zulkernine, 2018; Qiao & Zulkernine, 2020b), domain specific image processing (Li et al., 2019; Wojaczek et al., 2019), and multi-object detection (Gasmallah & Zulkernine, 2018; Qiao & Zulkernine, 2020b) and multilevel object recognition (Aggarwal, 2019). Some of the recent deep learning models have explored semantic image processing (Qiao & Zulkernine, 2020a), attention models to focus on specific features (Liu et al., 2018; Wang et al., 2018), and branching networks for hierarchical multilevel classification (Aggarwal, 2019).

* Corresponding author.
E-mail addresses: 15bak2@queensu.ca (B. Kolisnik), 14iach@queensu.ca (I. Hogan), farhana.zulkernine@queensu.ca (F. Zulkernine).

https://doi.org/10.1016/j.eswa.2021.115195
Received 6 November 2020; Received in revised form 31 March 2021; Accepted 9 May 2021
Available online 12 May 2021
0957-4174/© 2021 Elsevier Ltd. All rights reserved.

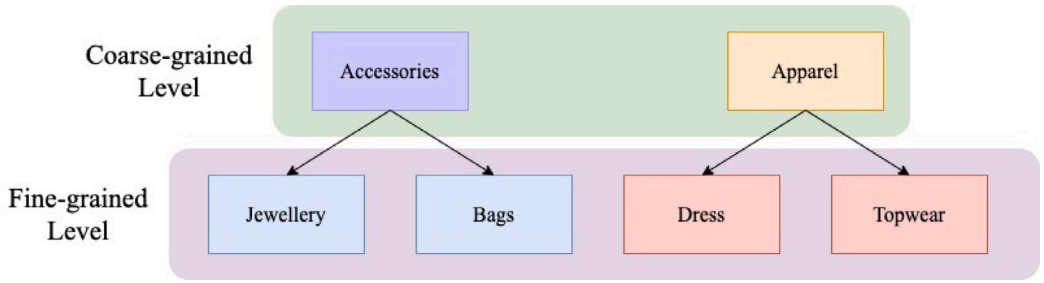

Fig. 1. Sample class hierarchy.

Hierarchical classification techniques have been applied to multiple domains including text categorization, protein function prediction, and music genre classification (Silla & Freitas, 2011). Hierarchical fashion image classification poses several unique challenges which are not necessarily present in other hierarchical classification domains. The folds of clothes, light conditions, variation in design, color and texture, quality of images taken using an automated camera or monitoring system, and the angle from which the photo is taken add to the existing challenges of fashion image classification (Hara et al., 2016; Zhu & Bain, 2017). In image classification problems, one object is prominent in the image, and therefore, features are extracted from the whole image to determine the class of the image. In hierarchical image classification, the object can have multiple labels defined in a hierarchy and all the labels must be recognized for the image.

Although image classification has been explored widely (Li et al., 2019; Wang et al., 2018), only a few approaches address the hierarchical multi-label image classification problem. With the rise in big data, multi-label image data sets are becoming more commonplace where one image can have multiple labels (Aggarwal, 2019) or hierarchical class labels (Planet, 2017). In this study, we consider a three-level class hierarchy represented as a tree to design our proposed model where the set of classes at each level are mutually exclusive and each child class has a single parent class. Coarse-grained classes do not have parents and represent the most high level abstract ideas. Lower level classes in the class hierarchy are referred to as fine-grained classes and the leaf nodes represent the most specific fine-grained classes. A visual representation of a sample class hierarchy is presented in Fig. 1 where the coarse-grained set of classes contain accessories and apparel, and the fine-grained set of classes contain jewelry, bags, dress and top-wear. In this example, jewelry and bags are the sub-classes of the accessories class. Similarly, dress and top-wear are the sub-classes of the apparel class.

While many CNN-based image classification approaches exist, only a few of these models are able to simultaneously predict multiple classes pertaining to a single image. Many existing techniques focus on improving segmentation performance (Chen et al., 2015; Hara et al., 2016; Li et al., 2019; Wang et al., 2018), or the robustness (Chen et al., 2015; Corbière et al., 2017; Liu et al., 2016a) of flat classifiers. These models do not leverage the coarse-grained class predictions in any capacity to refine the fine-grained predictions. However, the hierarchical relationships have been successfully exploited to improve the classification accuracy in other work (Cho et al., 2019; Lu et al., 2017; Seo & shik Shin, 2019). One of the best existing method for exploiting the hierarchy, branching convolutional neural networks (B-CNN), was shown to be more accurate in classifying a modified CIFAR-100 data set (Zhu & Bain, 2017) compared to multiple separate CNN models trained to predict each level of classes independently. However, the training algorithm of B-CNN, Branch Train, needs extensive training for tuning hyper-parameters.

The key contributions of our research are as follows.

- We propose a new model inspired by B-CNN, which we refer to as the Condition-CNN, for hierarchical image classification. Condition-CNN builds on B-CNN by first learning low-level features common to all levels in the class hierarchy through a common CNN base model. Then it learns the more specific features to distinguish the classes at each level in parallel through the use of separate convolutional sub-networks. Given that the classes at each level are mutually exclusive and each child class has a single parent class, Condition-CNN feeds the higher-level predictions as estimated priors to the lower level as input features to expedite the training and improve the prediction of the fine-grained classes. One of the applications of the model can be identifying clothing for e-commerce sites for automatic labeling, trading, sorting or analysis.
- Condition-CNN eliminates the Branch Train algorithm and its need for extensive training by implementing a novel Teacher Forcing algorithm, which greatly reduces the training time and allows faster convergence of the network. The course-grained predictions of the higher-level are input as prior information into the classification layers of the other branches performing lower-level fine-grained predictions. This is opposed to the hierarchical image classification approaches used by B-CNN and other multi-label CNNs that predict each label independent of the other labels.
- We validate Condition-CNN using the Kaggle Fashion Product Images data set (Aggarwal, 2019), which is composed of 41,027 images, and has a rich 3-class hierarchy compared to the other existing data sets. To compare the performance of the proposed Condition-CNN model, we implemented the B-CNN model and three separate baseline CNN models to independently predict the 3-levels of class labels. The experiments show that the estimated priors and the specific feature extraction branches of Condition-CNN contribute to a significant increase in its classification accuracy over the baseline models and the B-CNN. Condition-CNN achieved an accuracy of 99.8% in predicting Level 1 class labels, 98.1% for Level 2, and 91.0% for Level 3 class labels.

The rest of the paper is organized as follows. Section 2 presents an extensive review of the related work on fashion image classification. Model architectures of the B-CNN and the Condition-CNN including the proposed Teacher Forcing algorithm are described in Section 3. Implementation details are explained in Section 4. Section 5 illustrates the experiments and discusses the validation results. Finally, Section 6 concludes the paper with a summary and future work.

2. Related work

Fashion recognition is a diverse problem. Most existing approaches primarily seek to address object detection, object classification, and robustness (Chen et al., 2015; Li et al., 2019; Liu et al., 2016a). For each of the related experiments, the data set chosen allows the researchers to focus on one of these problems. Since we focus on hierarchical relationships between articles, the data used in this study does not require localization (Chen et al., 2015) of the garments. This section

covers a review of research on clothing localization and detection (Chen et al., 2015; Hara et al., 2016; Li et al., 2019; Wang et al., 2018), and leveraging semantic (Chen et al., 2015; Corbière et al., 2017; Liu et al., 2016a) and hierarchical nature of clothing categories in fashion image and object classification (Cho et al., 2019; Lu et al., 2017; Seo & shik Shin, 2019).

2.1. Object detection

For detecting clothes as objects worn by people, solutions typically attempt to identify the important regions for each piece of clothing to aid classifications (Hara et al., 2016; Li et al., 2019; Wang et al., 2018). Localization is performed by either selecting regions in bounding boxes which contain an article of clothing (Hara et al., 2016), or by masking unimportant parts of the images using attention mechanisms (Li et al., 2019; Wang et al., 2018). These techniques often augment the data with other information about the image to aid classification (Hara et al., 2016). The branch of research that focuses on object localization and classification techniques frequently use images of clothing worn by models provided in the DeepFashion data set (Cho et al., 2019; Corbière et al., 2017; Li et al., 2019; Liu et al., 2016a; Lu et al., 2017; Wang et al., 2018).

Wang et al. (2018) propose an attention based object localization approach. The model uses a convolution network, intermediate attention layers which select important regions and landmarks, and a dense multi-class classification network to create a direct relation between an image, the clothing category and its attributes. While the authors do not explore the effect of adding the intermediate layers independent of the other layers, the overall network performs well suggesting that these additions should be considered in future work. The paper is benchmarked on DeepFashion-C (Liu et al., 2016a) and achieves a top-3 classification accuracy of 90.99% and a top-5 classification accuracy of 95.78%.

Li et al. (2019) propose a refined implementation of attention localization. The model presents a two-stream network that creates a landmark heatmap which is multiplied with the original image to effectively create an attention mechanism. The altered image is then classified using a straight forward convolutional multi-class classification network. This technique is also benchmarked on the DeepFashion-C data set (Liu et al., 2016a). It reports a top-3 classification accuracy of 93.01% and a top-5 classification accuracy of 97.01%.

2.2. Semantic image classification

Semantic image classification takes into account the context i.e., the other objects in the image (Chen et al., 2015; Lu et al., 2017; Seo & shik Shin, 2019). Adding the typical semantic relationships between types of clothing to the feature set has proven to be a useful strategy for improving the accuracy of classification models (Chen et al., 2015; Lu et al., 2017; Seo & shik Shin, 2019). Real life labeled data is hard to obtain from industry applications and have worse quality than the benchmark data sets where images are centered nicely, have less noise and good light conditions (Chen et al., 2015), and are nicely labeled with all of the target classes (Corbière et al., 2017). Corbière et al. (2017) propose a training protocol for classifying e-commerce products using a weakly-supervised fine-grained labeled data set. The technique computes similarity between the images and the word embeddings, and only propagates error where labels are present. The model is tested but not exclusively trained on DeepFashion (Liu et al., 2016a), and achieves a top-3 category accuracy of 86.30% and a top-5 category accuracy of 92.80%.

Image segmentation is often used in such approaches to separate fashion articles. Hara et al. (2016) assign bounding boxes for image detection and garment localization. They propose a new technique to leverage object detection and use pose information to augment the data. The technique first uses a CNN to find candidate bounding boxes and features of these boxes. A Support Vector Machine (SVM) is then used to determine the most important features for each class. The pose information is then appended to the results of the SVMs, and finally a non-maximum suppression is used to discourage classification of conflicting garments. The model is evaluated on the Fashionista data set achieving average recall scores which range from 62.5% to 98.8% depending on the garment. The average recall across garments is 86.7%. The model, however, does not address hierarchical multi-label classification problems.

Chen et al. (2015) explore co-training a network using high quality data in addition to more realistic data to improve classification accuracy for the real life data sets. The model extracts region proposals and image features using a R-CNN model for segmenting the images into related feature regions. It has an alignment cost layer for finding correlations between the clean and messy image examples, a merging layer which takes the largest recognition of each feature between the two images, and two fully connected layers for the final multi-label attribute classification. The research demonstrates that existing clothing recognition techniques trained on clean data can be adapted to be used on real life data. In addition to the model, the authors construct two new, fine grained labeled data sets for testing clothing recognition: an online shopping data set and a street clothing data set. The street clothing data set contains the Fashionista data set with added fine grained labels and a set of 8,000 street fashion images for validation. The model is benchmarked on the street clothing data set achieving an accuracy of 48.32%, 15.3%, and 72.45% for clothing type, color, and pattern respectively. On the validation street-clothing data set, the model achieves 55.12%, 18.87%, and 75.90% on the same categories of classification.

2.3. Hierarchical classification

Silla and Freitas (2011) outline the different strategies for approaching hierarchical relationships in classification problems. Many approaches choose to ignore dependencies, opting for a flat classifier (Chen et al., 2015; Corbière et al., 2017; Hara et al., 2016; Li et al., 2019; Wang et al., 2018). The flat classification approach cannot predict multiple class labels for the same object and require many more neurons at the output layer compared to the ones that apply per level classification. In the per level classification approach, each level of granularity is handled by a separate classifier which receives information from the previous level (Seo & shik Shin, 2019; Zhu & Bain, 2017). Coarse-grained classes refer to classes which can be divided further into sub-classes while fine-grained classes do not have sub-classes.

Branching CNNs, an extension of the CNN architecture, can make predictions over a class hierarchy. Since clothing in broader categories (i.e., tops, bottoms) often share structural similarities, many techniques for fashion recognition aim to exploit these existing relationships by forming hierarchical model structures (Seo & shik Shin, 2019). Zhu and Bain (2017) introduce Branch-CNN (B-CNN), the most well-known branching CNN for hierarchical image classification. The authors note that the CNN layers learn increasingly abstract feature representations in a hierarchical manner from the input data, and thereby, evaluate the effectiveness of their model in terms of accuracy in predicting a class hierarchy of an image in descending order of abstraction. The B-CNN model achieved higher accuracy than the traditional CNNs on a collection of data sets such as MNIST (base 99.27%, B-CNN 99.40%) and CIFAR-100 (base 62.92%, B-CNN 64.42%). As these data sets only had one level and each image had a single class label, the authors synthesized various levels of classes to create a full class hierarchy for each image. The lowest level leaf nodes were assigned the true classes native to the data set, and the intermediate and root classes were created at the discretion of the authors for each data set. A custom training algorithm known as Branch Train is also proposed where weights assigned to the loss functions of different branchnets are manually adjusted during training. Initially, a greater emphasis is

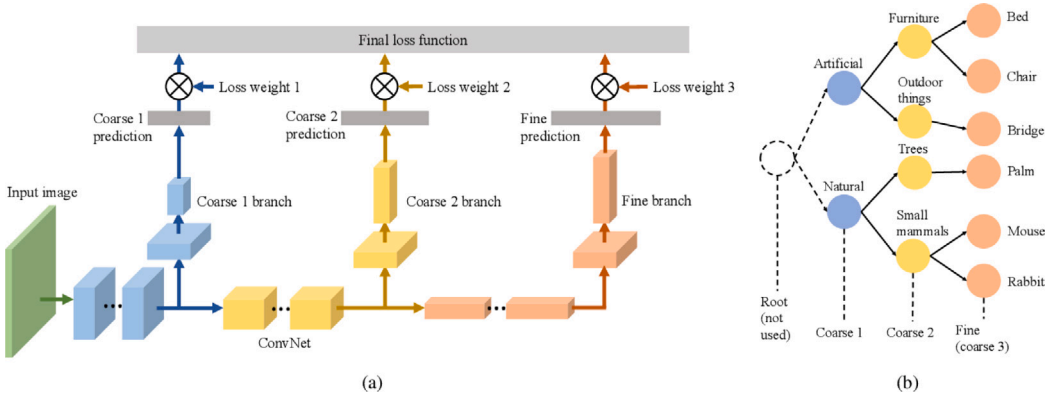

Fig. 2. B-CNN architecture (Zhu & Bain, 2017).

placed on the coarse-grained class level outputs by assigning a higher loss function weight to the coarse-grained loss than the other losses. These hyper-parameter loss weights are adjusted over time to place a greater emphasis on the fine-grained loss.

The Hierarchical Convolutional Neural Network (H-CNN) (Seo & shik Shin, 2019), another branching CNN, was later proposed based on the B-CNN model. H-CNN was the first attempt at using a branching CNN to predict a class hierarchy for a fashion image data set. The authors of H-CNN modified the Fashion-MNIST data set to represent a class hierarchy and found that H-CNN outperformed the baseline CNN models on the most specific class level in the class hierarchy. H-CNN is nearly identical to B-CNN in terms of the architecture and uses the same training algorithm with pre-chosen epoch indices. Thus, H-CNN suffers from the same training issues as B-CNN. H-CNN can be considered as a specific implementation of B-CNN designed for fashion images having three prediction branches. The proposed Condition-CNN model advances H-CNN and B-CNN by replacing the training algorithm and leveraging the architecture to achieve better performance.

Cho et al. (2019) propose that one of the key benefits of branching-style networks is that it gradually reduces the number of classes that must be differentiated at the classification layer. Their proposed model aims to extract the features for all levels of granularity in a base network, which are passed on to each branch. In addition, the features extracted for predicting a higher level class label are passed on to the branches that predict the lower or more fine grained class labels, allowing the network to share features for classification. This model is benchmarked on DeepFashion-test (Liu et al., 2016a) and achieved a top-3 accuracy of 91.24% and a top-5 accuracy of 95.68%.

In cases where hierarchical class structures are not readily available, it is advantageous to learn a structure. Lu et al. (2017) propose an algorithm for growing a deep neural network to compute object classes using hierarchical dependencies. They demonstrate that their technique is able to effectively create a hierarchical structure for many different problems including fashion classification. On the DeepFashion data set (Liu et al., 2016a), their model achieves a top-3 accuracy of 83.24% and a top-5 accuracy of 90.39%.

The proposed Condition-CNN uses the B-CNN as the base model and extends it for hierarchical multi-label classification of fashion images. Condition-CNN leverages the model architecture to use a Teacher Forcing training algorithm that eliminates the shortcoming of the Branch Train algorithm used in B-CNN and H-CNN. The Teacher Forcing algorithm applies conditional probability to learn class relationships and use the same for hierarchical classification. Conditional probability has been used for hierarchical classification in the existing literature (Pham et al., 2021; Taoufiq et al., 2020), but instead of learning the relationships as conditional probability, they compute the probability directly from the training data set. The next section presents the architecture of the Condition-CNN model. Since a new rich hierarchical multi-label fashion image data set has been published on Kaggle, we use this data set to validate the model as illustrated in Section 4.

3. Model description

We first present the architecture of B-CNN based on which Condition-CNN was designed. Next we describe the architecture of the proposed Condition-CNN model.

3.1. B-CNN

B-CNN is composed of a base CNN and multiple branchnets (Zhu & Bain, 2017) as can be seen in Fig. 2. It extends the VGG16 (Simonyan & Zisserman, 2015) for hierarchical image classification. Each branchnet, as shown in Fig. 3, receives extracted common low-level features from the base CNN as input and performs per level local classification for each level of granularity. The final layer of the base CNN feeds the feature outputs to the most specific branchnet corresponding to the fine-grained classes. The output predictions are ordered from coarse to fine-grained along concatenated convolutional layers corresponding to the hierarchical structure of the target classes. The abstract classes are predicted first and the more specific classes pertaining to the images are predicted next. Therefore, together the branchnets predict all the labels of the class hierarchy for a single image as can be seen in Fig. 2(b).

B-CNN uses a custom training algorithm Branch Train, which takes a large number of epochs to achieve a high accuracy for the fine-grained classes. The predictions of the branchnets are highly dependent on the shared layers. Backpropagation from the most fine-grained branchnet can alter every single convolutional block throughout the entire network and affect the future predictions of the coarse-grained branchnets. This is why the authors initially place an emphasis on optimizing the loss of the coarse-grained branchnet in their Branch Train algorithm so that the layers for the coarse-grained labels may converge first. In experimenting with B-CNN, we observed that when the initial emphasis was placed entirely on the fine-grained branchnet loss, updates to the weights caused the accuracy of the other branchnets to degrade and increased the convergence time.

CIFAR-100 data set that has only one level of class labels, is modified by the authors by synthesizing various levels of classes to create a 3-level class hierarchy for each image for validating the B-CNN model. A major difficulty of B-CNN (Zhu & Bain, 2017) is configuring the training algorithm, Branch Train, differently for each data set. In the B-CNN model the image features used for classification in the most fine-grained branch are derived entirely from the image features used to predict the coarse-grained classes. In order to train B-CNN, the loss function must be dynamically adjusted to shift emphasis from the coarse-grained predictions to the fine-grained predictions in the later epochs. The authors provide epoch indices to denote when these shifts should occur for the data sets used in their experiments, but learning the epoch indices hyper-parameters requires a considerable amount of experimentation for each new data set. Shifting the emphasis of the loss function too soon towards the loss of the most fine-grained prediction branch, can degrade the prediction accuracy of the coarse-grained classes.

Table 1
Comparison of related work on classification of fashion images.

Reference	Problem addressed	Data used	Accuracy%
Liu et al. (2016a)	**FashionNet**: Baseline CNN Model	DeepFashion	Top-3: 82.58 Top-5: 90.17
Corbière et al. (2017)	Poor/inconsistent data labels	DeepFashion	Top-3: 86.30 Top-5: 92.80
Wang et al. (2018)	Classification accuracy and attention	DeepFashion-C	Top-3: 90.99 Top-5: 95.78
Li et al. (2019)	Classification accuracy and attention	DeepFashion-C	Top-3: 93.01 Top-5: 97.01
Hara et al. (2016)	Detection using pose information	Fashionista	mAP: 31.1
Chen et al. (2015)	**DDAN-U**: Domain adaptation	Streetdata	Type-T1: 48.32 Color-T1: 15.34 Pattern-T1: 72.45
Lu et al. (2017)	Adapting network structure to implicit hierarchies in the data	DeepFashion	Top-3: 83.24 Top-5: 90.39
Cho et al. (2019)	Hierarchical classification	DeepFashion	Top-3: 91.24 Top-5: 95.68
Zhu and Bain (2017)	**B-CNN**: Hierarchical classification	Modified MNIST and CIFAR-100	Base 99.27, B-CNN 99.40 Base 62.92, B-CNN 64.42
Seo and shik Shin (2019)	**H-CNN**: Hierarchical classification	Fashion-MNIST	Top-1: 93.33

3.2. Condition-CNN

The main structural and functional properties of Condition-CNN that improve its performance compared to B-CNN with regard to hierarchical multi-label classification are summarized below.

- Similar to B-CNN, Condition-CNN uses a VGG16 base for the implementation as this is the most used basic CNN architecture. However, it uses a shallower base CNN as shown in Fig. 3 to extract the common image features that are fed to the branchnets. As a result, more of the trainable parameters are specific to each branchnet. This makes Condition-CNN more resilient to weight updates during backpropagation since error correction pertains to the specific prediction branch. It converges in fewer epochs and provides a greater accuracy compared to B-CNN.
- Both the Condition-CNN and the B-CNN implementations have three prediction branches to classify images having a 3-level class hierarchy. Each branch corresponds to one of the three levels of the class hierarchy of the Kaggle Fashion Product Images data set.
- Instead of the complex dynamic loss function that is used by the Branch Train algorithm in B-CNN, a categorical-cross entropy loss function is used in every branch of the Condition-CNN.
- Borrowing ideas from recurrent network models, at training time we apply a Teacher Forcing algorithm at the training phase to provide the ground truth priors to each subsequent prediction branch corresponding to a lower level in the class hierarchy. This enables faster convergence than using the higher level estimates as it allows for all of the prediction branches to be trained in parallel. If the Level 3 prediction branch makes an error, the gradients will not flow back to the Level 2 and Level 1 prediction branches. Only the Level 3 prediction branch and the common CNN base will be updated. This is a prominent feature that greatly reduces the number of epochs needed to train Condition-CNN and expedites convergence during the training phase.
- The Teacher Forcing training algorithm requires different training and testing architecture of the Condition-CNN.
- We experiment with two versions of Condition-CNN as illustrated in the validation section to demonstrate the performance of two different versions of the learned conditional probability matrix to be able to select the better one.

3.2.1. Model architecture

Condition-CNN uses the first four blocks of the original VGG16 network to create a modified VGG16 base CNN network. In the original VGG16, the third and fourth blocks have three pairs of convolutional and BatchNorm layers but in Condition-CNN, each of these blocks has only two pairs of convolutional layers, two BatchNorm layers and a MaxPool layer. The convolutional layers use three kernels for each filter to match the three color channels in the image. The kernel sizes and the number of filters differ depending on the layer and are illustrated in Fig. 3. Each MaxPool layer uses a 2×2 kernel with (2,2) stride. Furthermore, the third and fourth blocks are moved into the prediction branches to reduce the total number of parameters to be trained for the general base CNN part and allocate more parameters to learn features specific to the lower levels. Our experiments demonstrate that the above changes allow for superior classification accuracy across the class hierarchy for Condition-CNN compared to the B-CNN.

Condition-CNN multiplies the higher level class predictions by a Conditional Probability Weight Matrix (CPWM) to obtain estimates for the lower level predictions, which is why we call our model 'Condition-CNN'. The estimates are then added to the extracted feature vector to compute the input to the softmax function that generates the final class prediction as the output. Our training and inference definitions of Condition-CNN can be seen in Figs. 4 and 5 where the $\times 3$ and $\times 2$ represent blocks that are repeated in the network. In these figures the + layers represent vector addition, the Dropout layers all have a dropout rate of 0.5, the ground truth cells represent the ground truth as one hot class probability vectors, and all dense layers have a bias.

3.2.2. Learning algorithm

Condition-CNN learns associations between the sub-classes and the parent classes within the class hierarchy by modeling the conditional

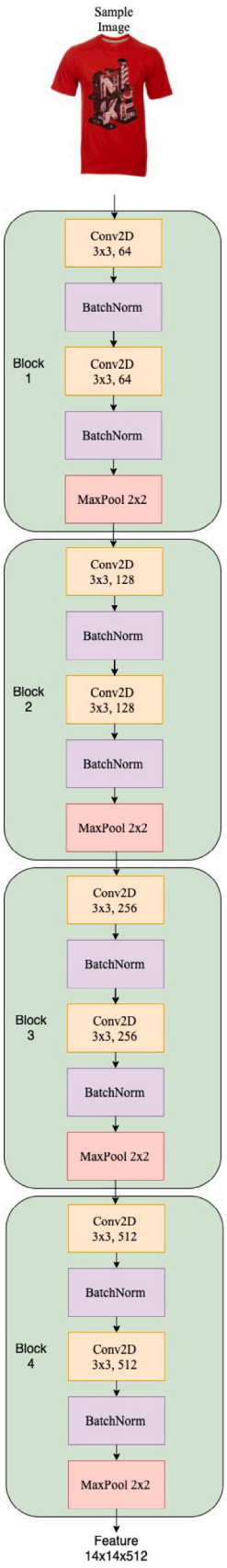

Fig. 3. Base CNN model modified from VGG16.

probabilities as weights. Some work has been done that apply conditional probability in image classifiers (Pham et al., 2021; Taoufiq et al., 2020). Most of these work exploit the relationships among diseases in classification of chest X-rays (Pham et al., 2021) and urban structures (Taoufiq et al., 2020). Instead of letting the model learn the conditional probability, it is computed based on the given data before the training phase, which is then fed to the classification step. For the clothing data used in this study, the class labels at each level in the hierarchy are mutually exclusive. This means the sample space for a class of the fine-grained label can be partitioned based on the estimated priors and the learned conditional probabilities. This is known as the law of total probability as shown in Eq. (1).

$$P(A) = \sum_{i=1}^{n} P(A|B_i)P(B_i) \tag{1}$$

To learn associations between the classes at different levels in the label hierarchy, Condition-CNN learns a CPWM as seen in Eq. (2) and (3). Considering that there are 3 levels in the class hierarchy, we refer to Level 1 as the most coarse level and Level 3 as the most fine-grained class label.

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots \\ \vdots & \ddots & \\ w_{I1} & & w_{IJ} \end{bmatrix} \tag{2}$$

$$W = \begin{bmatrix} P(A = a_1|B = b_1) & P(A = a_2|B = b_1) & \dots \\ \vdots & \ddots & \\ P(A = a_1|B = b_I) & & P(A = a_J|B = b_I) \end{bmatrix} \tag{3}$$

where $P(A = a_j)$ is the probability that the fine-grained class is j, $P(B = b_i)$ is the probability that the coarse-grained class is i and $P(A = a_j|B = b_i)$ is the conditional probability that the fine-grained class is j given that the coarse-grained class is i. The shape of this matrix is $I \times J$ where I is the number of coarse-grained classes and J is the number of fine-grained classes. Each element of this weights matrix is constrained such that $0 \leq P(A = a_j|B = b_i) \leq 1$ and additionally $(\sum_{j=1}^{J} P(A = a_j|B = b_i)) = 1$ for each row. This learned CPWM exists between the Level 1 and Level 2 prediction branches as well as the Level 2 and Level 3 prediction branches.

We treat the predictions of all but the lowest level prediction branch as estimated priors. The priors are the outputs of the softmax function from a parent level prediction branch. The softmax output is vector $\mathbf{v}$ of size $1 \times I$, which represents the probability distribution of the parent level coarse-grained classes.

$$\vec{\mathbf{v}} = \begin{bmatrix} P(B = b_1) & P(B = b_2) & \dots & P(B = b_I) \end{bmatrix} \tag{4}$$

The prior probability vector is then multiplied by the learned CPWM for the immediate lower level prediction branch as shown in Eq. (5).

$$\vec{\mathbf{p}} = \vec{\mathbf{v}}W \tag{5}$$

Using the definition of W from Eq. (3) and $\vec{\mathbf{v}}$ from Eq. (4), the resulting row vector $\vec{\mathbf{p}}$ can be written as shown in Eq. (6).

$$\vec{\mathbf{p}} = \begin{bmatrix} \sum_{i=1}^{I} P(A = a_1|B = b_i)P(B = b_i) & \dots & \sum_{i=1}^{I} P(A = a_J|B = b_i)P(B = b_i) \end{bmatrix} \tag{6}$$

By applying the law of total probability from Eq. (1) to each element of $\vec{\mathbf{p}}$, we derive $\vec{\mathbf{p}}$ as the vector of probabilities for the fine-grained classes as in Eq. (7).

$$\vec{\mathbf{p}} = \begin{bmatrix} P(A = a_1) & P(A = a_2) & \dots & P(A = a_J) \end{bmatrix} \tag{7}$$

In our reference implementation, we multiply the softmax output of our Level 1 prediction branch by the CPWM for our Level 2 prediction branch as shown in Eq. (5). Similarly, we multiply the softmax output of our Level 2 prediction branch by a CPWM for our Level 3 prediction branch. Each prediction branch in Condition-CNN uses a categorical cross-entropy loss function.

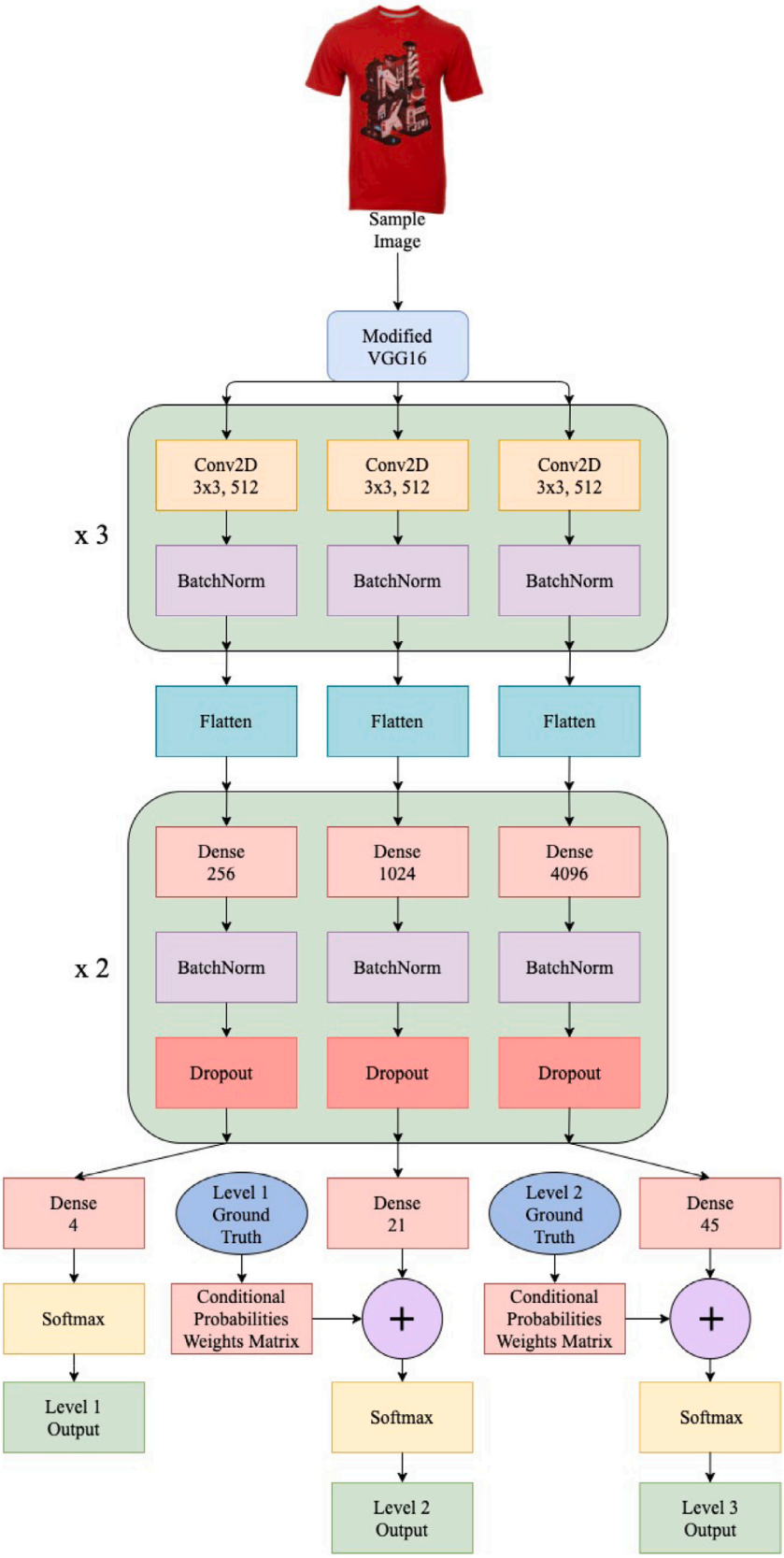

Fig. 4. Training definition of Condition-CNN.

Since these are probability estimates for the coarse-grained classes, at inference time, we proceed with the notation as shown in Eq. (8) to represent the probability estimates for the fine-grained classes.

$$\hat{\mathbf{p}} = \left[P(\widehat{A = a_1}) \quad P(\widehat{A = a_2}) \quad \dots \quad P(\widehat{A = A_J}) \right] \tag{8}$$

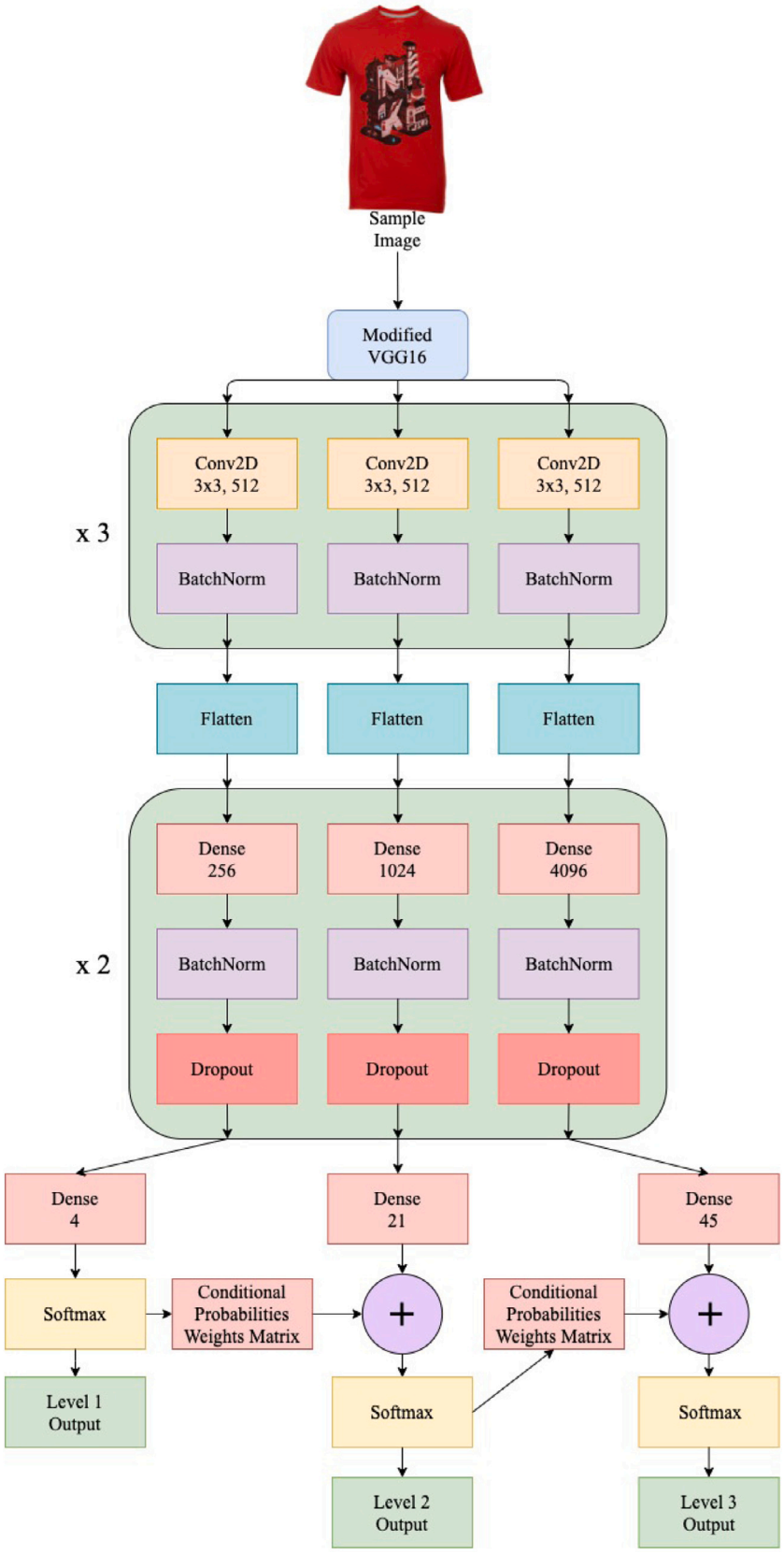

Fig. 5. Inference definition of Condition-CNN.

Next we describe the model implementation and validation results.

4. Implementation

The architecture of the base CNN (modified VGG16) model (Fig. 3), the training (Fig. 4) and the inference models (Fig. 5) are explained in Section 3. As explained in the architecture, due to the application of the Teacher Forcing algorithm, Condition-CNN requires a different training and inference architecture. This is due to the fact that during training the ground truth information is available to inform the lower-level predictions whereas during testing the higher-level predictions substitute the ground truth. As a result, we implemented two versions of the Condition-CNN to correspond to the training and inference architecture and the functional requirements. Data was preprocessed and used to train and validate the model. During the training phase, the model learns the CPWM using the training data set that allows efficient and accurate prediction of class labels of the testing data set during the inference phase. In this section, we first describe the data, illustrate the training of Condition-CNN, and present the experimental results obtained from both the training and testing phases. The results demonstrate the superiority of the proposed Condition-CNN compared to the B-CNN.

4.1. Implementation environment

The OS used in the implementation environment was Ubuntu 18.04, with a 12 Core Intel Xeon Gold 6130 CPU @ 2.10 GHz, 24GB of RAM, and a Tesla V100 GPU with 32 GB of VRAM. Our code was written in Python 3.6.3 and Keras 2 using the Tensorflow 2.1 backend. The NVIDIA CUDA driver was version 10.1.

4.2. Data preprocessing

The Kaggle Fashion Product Images data set (Aggarwal, 2019) is used in validating Condition-CNN for the following reasons.

- It is the largest available data set with a predefined class hierarchy and contains 41,027 images. The classes are described in a csv file. There are 3 levels in the class hierarchy. These are the coarse-grained masterCategory level which contains 4 classes, the fine-grained subCategory level which contains 21 classes, and lastly the most fine-grained level articleType which contains 45 classes. An example image and its classes are shown in Fig. 6.
- The DeepFashion data set (Liu et al., 2016a) has been widely used by researchers for fashion classification research. However, DeepFashion does not have a defined class hierarchy.
- The hierarchical multi-label Fashion-MNIST and CIFAR-100 data sets that were created by other researchers, were not released and hence, could not be used in this study.

We explained the Teacher Forcing learning algorithm used by Condition-CNN in Section 3 where we referred to Level 1 as the most coarse-grained label and Level 3 as the most fine-grained label. Therefore, for the Kaggle Fashion Product Images data set, masterCategory would map to Level 1, the fine-grained subCategory would map to Level 2, and the most fine-grained articleType would map to Level 3. Other data sets may have a different number of levels in the class hierarchy in which case the number of branchnets can be adjusted accordingly for both B-CNN and Condition-CNN to match the number of levels in the class hierarchy.

We applied a number of transformations to preprocess the images. The images were rescaled so that the RGB values were between 0 and 1. Images were resized to 224 x 224, and randomly flipped horizontally, zoomed (up to 10%), and sheared (up to 10%) to add random noise for regularization and improve the classification accuracy. Since Condition-CNN uses BatchNorm layers, we applied sample-wise centering and standard normalization across the batch. The data was split into 65% train, 25% test and 10% validation set with the same distribution of classes. Condition-CNN benefits when the training and test data sets are similar in class distribution due to the learned condition weights matrix.

Fig. 6. Sample image with labels (Aggarwal, 2019).

All of the models trained in our experiments have their common VGG16 weights initialized from a pre-trained VGG16 model trained on ImageNet. We trained all of the models on the Fashion Product Images data set for 20 epochs. 20 epochs was chosen as all of the models' validation loss plateaued after 15 epochs. At the end of each epoch we computed the model's loss on the validation data set. After the 20 epochs we proceeded to test the model with the set of weights that achieved the lowest loss on the validation data set.

5. Validation and results

Condition-CNN was designed based on the B-CNN model for hierarchical multi-label classification of fashion images with a view to improving the training efficiency, classification accuracy, and the overall model performance. Therefore, the results presented in this section illustrate the functionality and validate the achievements of the objectives. First we present the results from training time, the CPWM, and illustrate its effectiveness in improving the prediction accuracy and reducing the training time. Next we compare the performance of Condition-CNN with B-CNN and three separate baseline CNNs, each of which predict a different level of classes.

5.1. Effectiveness of the CPWM

Condition-CNN implements a Teacher Forcing algorithm to learn the CPWM during the training phase. As explained in Section 3, to predict the class label for any level at the softmax layer, the class predictions of the upper levels are multiplied by the CPWM to estimate a probability of the target class label. The estimates are then added to the extracted feature vectors before feeding into the softmax inputs. During training time, instead of predicted class labels, the ground truth labels of the upper level are multiplied by the CPWM and backpropagation is used to iteratively adjust the CPWM values to reduce the error in classifying

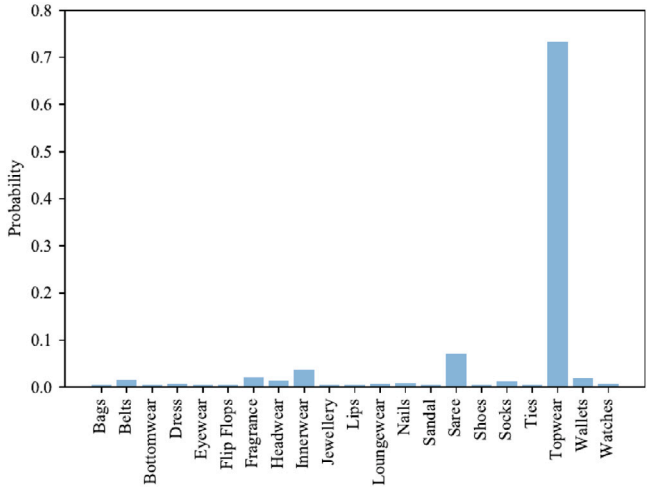

Fig. 7. Level 2 sample prediction distribution no sharpening.

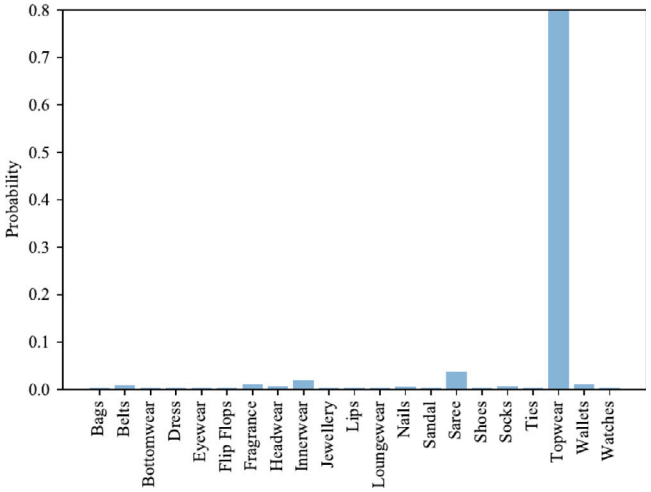

Fig. 8. Level 2 sample prediction distribution with sharpening.

the training data. The CPWM is initialized to 0's during training and the values change during the training phase.

5.1.1. Observation 1: Learning of CPWM

An example of the CPWM learned by the top performing Condition-CNN model is shown in Table 2.

5.1.2. Observation 2: Sharpening class prediction

Through experimental study, it was validated that using the estimated probabilities in addition to the feature vector extracted by the CNN layers of the prediction branch, gave a greater accuracy than using the feature vector alone. It was also found that the greatest performance came from adding the probability estimates $\hat{\mathbf{p}}$ to the dense feature vector before feeding it to the final softmax function as shown in Eq. (9).

$$Z_j = \widehat{P(A = a_j)} + x_j \tag{9}$$

$$P(A = a_j) = softmax(Z_j) = \frac{\exp Z_j}{\sum_{k=1}^{K} \exp Z_k} \tag{10}$$

$$P(A = a_j) = softmax(Z_j) = \frac{\exp \widehat{P(A = a_j)} \exp x_j}{\sum_{k=1}^{K} \exp \widehat{P(A = a_k)} \exp x_k} \tag{11}$$

The implications of using the probability estimates based on the coarse-grained class predictions can be seen in Eq. (10). We interpret Eq. (11) as re-scaling each outcome in the distribution of class predictions. The predictions are scaled based on the probability estimates derived from the coarse-grained prediction. This has the effect of "sharpening" the prediction of class distribution as shown in Fig. 8, compared to the one shown in Fig. 7 where only the dense feature vector was used. This approach accounts for the coarse-grained predictions and reduces the prediction probability of classes that are not feasible based on the coarse-grained estimates. Due to the nature of softmax, this simultaneously increases the probability of feasible classes that are sub-classes of the coarse-grained prediction.

These sample prediction distribution plots were generated using a fully trained Condition-CNN where the ground truth class was the Level 2 class Top-wear. Fig. 7 also shows the class probabilities of the other Level 2 classes, which are greater compared to that shown in Fig. 8. In Fig. 8, the Level 2 predictions are conditioned based on Level 1 predictions, and as a result, the probability estimates for the classes other than Top-wear have been greatly reduced. In this particular example, due to a coarse-grained parent class prediction of "Apparel" the probability estimate for "Topwear" increases from Fig. 7 to Fig. 8. The results demonstrate the effectiveness of the Teacher Forcing algorithm in improving the prediction accuracy of the Level 2 and Level 3 prediction branches.

$$CE = -\sum_{i}^{Classes} t_i \ln(f(Z_j)) \tag{12}$$

The categorical cross-entropy loss function applied to each prediction branch in Condition-CNN is shown in Eq. (12), where t_i is the true label, 0 or 1, for class i, f is the softmax function, and Z_j is the class score from Eq. (9). For our experiments we have used the categorical cross-entropy loss function for all of our baseline VGG16 models.

5.2. Evaluation of condition-CNN

To validate the superior performance of Condition-CNN, we implemented several other models to compare the performance.

- We trained three separate dedicated CNN baseline models based on the VGG16 architecture to independently predict the class labels for each level in the class hierarchy. We refer to these models as Level 1, Level 2, and Level 3 baseline models to predict class labels for the masterCategory, subCategory and articleType labels respectively.
- We also implemented the B-CNN model to compare model performances using the same Kaggle Fashion Product Images data set since the original data set used by the authors of B-CNN were not released. Both the B-CNN and Condition-CNN models have three prediction branches, one for each level of the class hierarchy in the data set. We also used the same number of layers and layer hyper-parameters for all of the models tested.

In our experiments we trained two versions of Condition-CNN, Condition-CNN and Condition-CNN B, both of which refer to the same Condition-CNN network but use different weight constraints to learn the conditional weights. For the CPWM (as shown in Table 2), Conditional-CNN enforces that the magnitude of each row of the CPWM is 1 whereas Condition-CNN B enforces that the sum of each row of the CPWM is 1. When enforcing the magnitude of 1, the CPWM does not correspond to a true probability distribution estimate but rather scaled probability estimates. However, this Condition-CNN implementation still performed well during our testing and we present the results for both. If true conditional probability estimates are desired, the Condition-CNN B model must be used.

Table 2
Example of a learned CPWM by the top-performing Condition-CNN.

	Bags	Belts	Bottomwear	Dress	Eyewear	Flip Flops	Fragrance
Accessories	0.436091	0.104293	0	0	0.073139	0	0
Apparel	0	0	0.118106	0.013426	0	0	0
Footwear	0	0	0	0	0	0.10557	0
Personal care	0	0	0	0	0	0	0.853525
	Headwear	Innerwear	Jewelry	Lips	Loungewear and nightwear	Nails	Sandal
Accessories	0	0.01241	0	0.018871	0	0	0
Apparel	0	0.077331	0	0	0.00731	0	0
Footwear	0	0	0	0	0	0	0.103845
Personal care	0	0	0	0.009158	0	0.137317	0
	Saree	Shoes	Socks	Ties	Topwear	Wallets	Watches
Accessories	0	0	0.021543	0.021501	0	0.082737	0.229417
Apparel	0.001784	0	0	0	0.782043	0	0
Footwear	0	0.790585	0	0	0	0	0
Personal care	0	0	0	0	0	0	0

Table 3
Branch Train training schedule.

Epoch	Level 1 loss weight	Level 2 loss weight	Level 3 loss weight
0	0.98	0.01	0.01
5	0.1	0.8	0.1
10	0.1	0.2	0.7
15	0	0	1

Table 4
Comparison of the top-1 test accuracy after training models for 20 epochs and selecting the best weights using a validation data set.

Model	Accuracy%			Trainable parameters
	Level 1	Level 2	Level 3	
Level 1 Baseline	**0.997949**			37,529,924
Level 2 Baseline		0.977637		115,608,405
Level 3 Baseline			0.899707	439,799,661
B-CNN	0.991797	0.923828	0.776172	752,176,774
Condition-CNN	0.997559	**0.980664**	**0.910449**	583,560,587
Condition-CNN B	0.997656	0.977344	0.904492	583,560,587

5.2.1. Observation 1: Speed of model convergence

B-CNN uses the Branch Train algorithm and it requires assignment of training importance by loss weights to the different branchnets. The loss weight schedule for the Branch Train algorithm used in B-CNN is outlined in Table 3 where Epoch 0 shows the initial set of loss weights. Initially the Level 1 branchnet is given the most importance to train its hyperparameters. The weights for Branch Train are adjusted at the end of each epoch as listed in the table. For example, starting at epoch 5 the loss weights for computing the weighted average of the losses are adjusted. The Level 1 loss weight becomes 0.1, the Level 2 loss weight becomes 0.8, and the Level 3 loss weight becomes 0.1, to shift the emphasis on the Level 2 loss.

In practice the training time is difficult to compare due to the training strategy used in Branch Train for performance improvement. However, if we consider the speed of training and validation for the fine-grained classes, Condition-CNN is able to achieve a validation accuracy of 0.9 after only 7 epochs rather than 14 epochs as required by B-CNN. This represents a training speedup of up to 50% for the most specific classes on the 3 tiered problem. The fine-grained classes are most effected by the propagation of errors. Hence we present the comparative performance of the Level 3 branchnet for the two models. This performance gap could be exacerbated on tasks with more layers in the hierarchy.

We present the loss plots for each of the models in Fig. 9. The plots show the training and validation loss for the baseline CNNs, B-CNN, Condition-CNN, and Condition-CNN B. As expected, the Level 1 loss decreases much faster than Level 2 and Level 3 loss. Level 1 loss converges slightly faster in B-CNN because greater emphasis is placed on that in the beginning than in Condition-CNN. However, it is evident that the Level 2 and Level 3 losses decrease much faster for both training and validation phases in Condition-CNN compared to B-CNN and baseline CNNs.

5.2.2. Observation 2: Prediction accuracy and number of parameters

Fig. 9 also illustrates that both versions of Condition-CNNs achieve a lower loss or higher accuracy in predicting Level 2 and 3 classes faster during the training phases than the other models namely the B-CNN and the baseline CNNs. For example, after 5 epochs the training loss is around 0.6 for Level 2 and 1.4 for Level 3 in B-CNN, and 0.09 for Level 2 and 0.38 for Level 3 in the baseline CNNs. In contrast, after 5 epochs, Level 2 and 3 branchnets in Condition-CNN reach a loss of only 0.09 and 0.38 respectively. Therefore, for industry applications, Condition-CNN can give very good accuracy very fast. Although the individual CNNs gives comparable performance in terms of training time and accuracy, individual CNNs cannot predict multiple labels and cannot infer about the relationship with other levels. Implementing three CNNs require separate set up, training and validation effort compared to one Condition-CNN.

A comparison of the test accuracy of all of the models is presented in Table 1. Many work in the literature report top-N accuracy which implies that the correct class label is among the N most probable classes predicted by the model. If N is greater, the reported accuracy is therefore higher. We report top-1 accuracy in this study as the model predicts multiple labels at the same time and the prediction accuracy for the 3 levels vary within a range of 90% to 99.8%. As shown in the table, Condition-CNN achieves much greater accuracy than the other models for Level 2 and Level 3 class prediction.

It is also shown in the last column of Table 4 that the number of trainable parameters of Condition-CNN is much less than that of B-CNN and a little less than the total number of parameters of the three baseline CNN models.

5.3. Discussion

We observe that both versions of Condition-CNN achieved the highest accuracy on Level 3 in our test data set and matched the Level 1 Baseline accuracy to three decimal places. Additionally, Condition-CNN achieved the highest test accuracy on Level 2. This is despite having only 583,560,587 trainable parameters which is 77% of 752,176,774, the number of parameters required by B-CNN, and 98% of 592,937,990, the total number of parameters required by the three baseline CNN models. Furthermore, Level 3 baseline model requires about 4 times more parameters than the Level 2 and about 10 time more parameters than the Level 1 baseline models, which means that the number of parameters increases exponentially with the number of levels. Therefore,

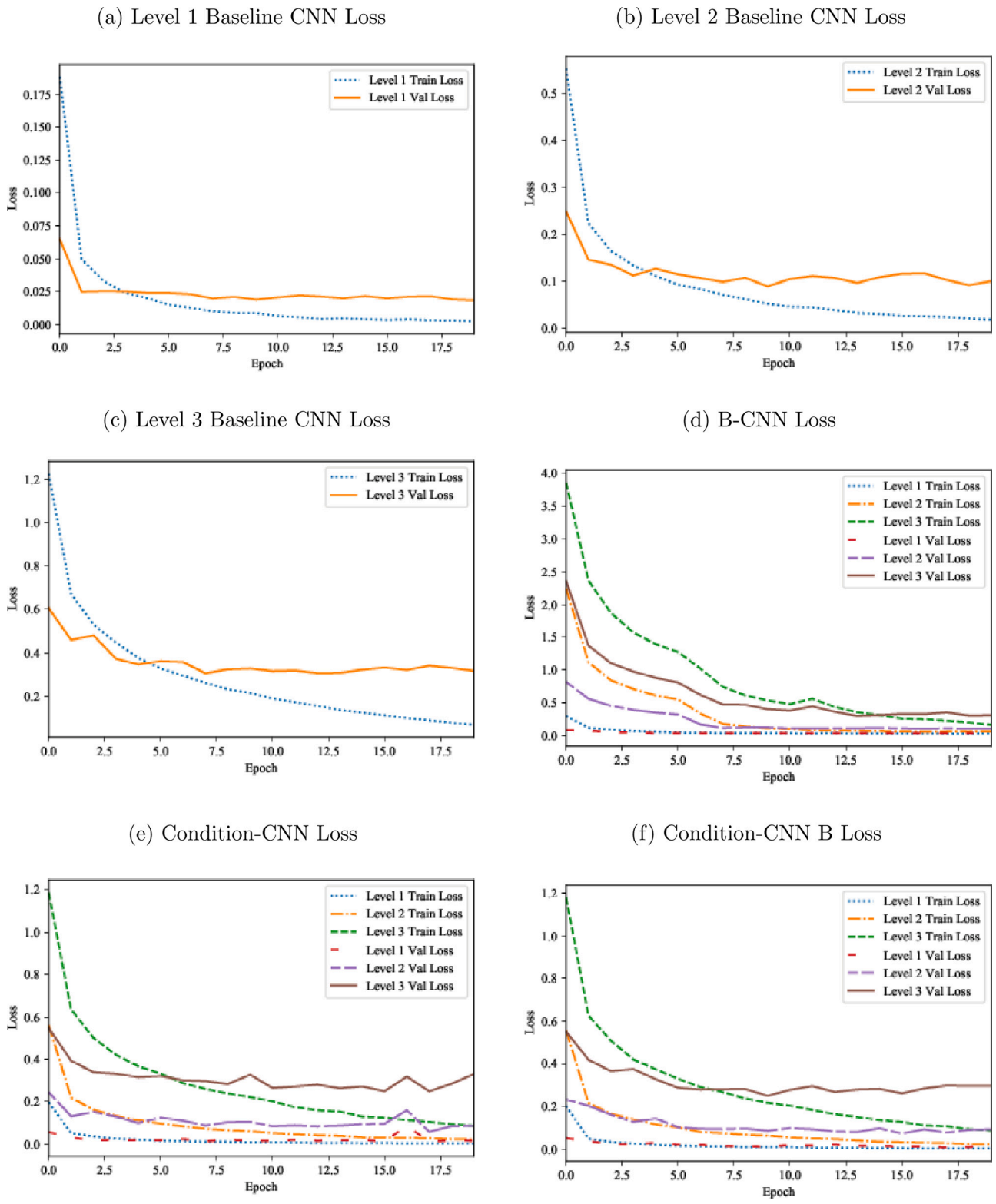

Fig. 9. Comparison of model loss.

for a larger hierarchy Condition-CNN will likely need fewer parameters than all the baseline models and the B-CNN model, which may be a topic of the future research.

The experiments provide empirical evidence that utilizing contextual information from the coarse-grained predictions can increase prediction accuracy of fine-grained labels. Condition-CNN provides multiple benefits for industry practitioners. Compute and training time may be saved by training a single Condition-CNN model rather than training multiple separate CNN models for each level. It also achieves excellent predictive accuracy across the label hierarchy using a common training strategy with no hyper-parameter testing as required by the Branch Train algorithm.

Fig. 9 demonstrates that the Condition-CNN converges much more quickly than B-CNN using both versions of the weight constraints.

This is very apparent for the Level 2 and Level 3 train and validation losses. For example, by the end of epoch 5, the Level 3 validation loss for Condition-CNN B is 0.287865 which is lower than the Level 3 baseline CNN validation loss of 0.361470 and much lower than B-CNN's validation loss of 0.810750. Similarly, validation loss after 5 epochs for Level 2 is 0.103015 for Condition-CNN B, 0.114130 for baseline CNN, and 0.322468 for the B-CNN. This observation proves that Condition-CNN is much more suitable for industry application where fast performance and high prediction accuracy at each level of class hierarchy is important.

One important thing to note is that although the separate baseline CNNs show comparable but slightly worse performance than the Condition-CNN, they do not provide the inter-class relationship or feature dependence information for decision making. Each CNN is trained independent of the other CNNs and one level's prediction cannot provide any information about the other levels' predictions. In contrast, in Condition-CNN, Level 1 and 2 branchlets can still give a correct prediction when Level 3 prediction is incorrect. Therefore, we can still obtain partial prediction about the higher level classes for an image.

6. Conclusion and future work

Human cognitive system extracts and classifies patterns in a hierarchical manner, which reduces of ambiguity, allows better and faster correlation learning in multiple levels, and thereby, enables intelligent decision making. The traditional approaches to image recognition considers all class labels belonging to a single flat level. Recent deep learning models extract patterns through multiple layers and combines fine-grained features to more coarse-grained features for image or object detection. In this study, we propose a novel hierarchical multi-label classification model, Condition-CNN, which builds upon the existing branching neural network model B-CNN (Zhu & Bain, 2017) and demonstrates superior performance with regard to computational complexity, training time and classification accuracy. It requires only 77% of the total number of hyper-parameters required by B-CNN and achieved 99.8% accuracy for Level 1 classes, 98.1% for Level 2 classes, and 91.0% Level 3 classes in the increasing order of specificity (Level 1 coarse-grained and Level 3 is the most fine-grained class) on the Kaggle Fashion Product Images data set (Aggarwal, 2019) containing 3-levels of class the top row of Table 3] Condition-CNN applies a novel Teacher Forcing algorithm to learn a Conditional Probability Weight Matrix (CPWM) during training phase. It applies the CPWM to estimate class probability of the higher level during the scoring phase, which is used to inform the lower level predictions. The Teacher Forcing algorithm reduces the training time, especially for the most specific classes with existing classification dependencies. In the experiments presented, this speedup was up to 50%. The estimated higher level class probability and the extracted feature vectors help achieve a more accurate lower level prediction of the class labels. Compared to the existing approaches which apply multiple separate models to predict each level of class labels, Condition-CNN provides one solution and learns feature correlations among the different levels of classes in a class hierarchy. It converges much faster than B-CNN and provides a fast and efficient solution for the fashion industry.

Future work will focus on using different CNN networks as the base model in Condition-CNN to analyze the impact on the final prediction accuracy for each of the branchnets that predicts different levels of class labels in the class hierarchy. It may be worth experimenting other optimizations such as reducing the number of parameters in Level 1 feature extraction branch. We also intend to explore continuous learning paradigm for hierarchical multi-label image classification and for greater number of classes in the class hierarchy. It would be interesting to explore and apply object localization, detection and attention to hierarchical multi-label classification when multiple objects exist in an image.

CRediT authorship contribution statement

Brendan Kolisnik: Conceptualization, Methodology, Software, Validation, Formal analysis, Data curation, Writing - original draft. **Isaac Hogan:** Investigation, Writing - review & editing, Supervision. **Farhana Zulkernine:** Supervision, Funding acquisition, Project administration, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The research was funded by NSERC (Natural Sciences and Engineering Research Council) Canada and was done in collaboration with industry partner, AIdeal Inc.

References

Aggarwal, P. (2019). Fashion product images dataset. https://www.grandviewresearch.com/press-release/global-image-recognition-market [Dataset; accessed: 2020-06-18].

Bay, H., Tuytelaars, T., & Gool, L. V. (2006). SURF: Speeded up robust features. *Computer Vision-ECCV*, 404–417.

Chen, Q., Huang, J., Feris, R., Brown, L. M., Dong, J., & Yan, S. (2015). Deep domain adaptation for describing people based on fine-grained clothing attributes. In *2015 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 5315–5324).

Cho, H., Ahn, C., Yoo, K. M., Seol, J., & Lee, S. (2019). Leveraging class hierarchy in fashion classification. In *2019 IEEE/CVF international conference on computer vision workshop (ICCVW)* (pp. 3197–3200).

Corbière, C., Ben-Younes, H., Ramé, A., & Ollion, C. (2017). Leveraging weakly annotated data for fashion image retrieval and label prediction. In *2017 IEEE international conference on computer vision workshops (ICCVW)* (pp. 2268–2274).

Gasmallah, M. H., & Zulkernine, F. (2018). Video predictive object detector. In *2018 IEEE 9th annual information technology, electronics and mobile communication conference (IEMCON)* (pp. 365–371). http://dx.doi.org/10.1109/IEMCON.2018.8615054.

Grand View Research (2020). Image recognition market worth $109.4 billion by 2027. https://www.grandviewresearch.com/press-release/global-image-recognition-market [Online; accessed: 2020-06-18].

Hara, K., Jagadeesh, V., & Piramuthu, R. (2016). Fashion apparel detection: The role of deep convolutional neural network and pose-dependent priors. In *2016 IEEE winter conference on applications of computer vision (WACV)* (pp. 1–9). Los Alamitos, CA, USA: IEEE Computer Society, http://dx.doi.org/10.1109/WACV.2016.7477611, https://doi.ieeecomputersociety.org/10.1109/WACV.2016.7477611.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105.

Li, P., Li, Y., Jiang, X., & Zhen, X. (2019). Two-stream multi-task network for fashion recognition. In *2019 IEEE international conference on image processing (ICIP)* (pp. 3038–3042).

Liu, N., Han, J., & Yang, M. (2018). Picanet: Learning pixel-wise contextual attention for saliency detection. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 3089–3098). http://dx.doi.org/10.1109/CVPR.2018.00326.

Liu, Z., Luo, P., Qiu, S., Wang, X., & Tang, X. (2016a). Deepfashion: powering robust clothes recognition and retrieval with rich annotations. In *2016 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1096–1104).

Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., & Feris, R. (2017). Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Pham, H. H., Le, T. T., Tran, D. Q., Ngo, D. T., & Nguyen, H. Q. (2021). Interpreting chest X-rays via CNNs that exploit hierarchical disease dependencies and uncertainty labels. *Neurocomputing*, *437*, 186–194. http://dx.doi.org/10.1016/j.neucom.2020.03.127, https://www.sciencedirect.com/science/article/pii/S0925231221000953.

Planet (2017). Planet: Understanding the amazon from space. https://www.kaggle.com/c/planet-understanding-the-amazon-from-space/data [Dataset; accessed: 2020-06-19].

Qiao, D., & Zulkernine, F. (2020a). Dilated squeeze-and-excitation U-net for fetal ultrasound image segmentation. In *2020 IEEE conference on computational intelligence in bioinformatics and computational biology (CIBCB)* (pp. 1–7). http://dx.doi.org/10.1109/CIBCB48159.2020.9277667.

Qiao, D., & Zulkernine, F. (2020b). Vision-based vehicle detection and distance estimation. In *2020 IEEE symposium series on computational intelligence (SSCI)* (pp. 2836–2842). http://dx.doi.org/10.1109/SSCI47803.2020.9308364.

Seo, Y., & shik Shin, K. (2019). Hierarchical convolutional neural networks for fashion image classification. *Expert Systems with Applications*, *116*, 328–339. http://dx.doi.org/10.1016/j.eswa.2018.09.022, http://www.sciencedirect.com/science/article/pii/S0957417418305992.

Silla, C. N., & Freitas, A. A. (2011). A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, *22*(1), 31–72. http://dx.doi.org/10.1007/s10618-010-0175-9.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556.

Taoufiq, S., Nagy, B., & Benedek, C. (2020). Hierarchynet: Hierarchical CNN-based urban building classification. *Remote Sensing*, *12*(22), 3794. http://dx.doi.org/10.3390/rs12223794.

Wang, W., Wang, W., Xu, Y., Shen, J., & Zhu, S. (2018). Attentive fashion grammar network for fashion landmark detection and clothing category classification. In *2018 IEEE/CVF conference on computer vision and pattern recognition* (pp. 4271–4280).

Wojaczek, A., Kalaydina, R., Gasmallah, M., Szewczuk, M. R., & Zulkernine, F. (2019). Computer vision for detecting and measuring multicellular tumor shperoids of prostate cancer. In *2019 IEEE symposium series on computational intelligence (SSCI)* (pp. 563–569). http://dx.doi.org/10.1109/SSCI44817.2019.9002908.

Zhu, X., & Bain, M. (2017). B-CNN: Branch convolutional neural network for hierarchical classification. CoRR abs/1709.09890. http://arxiv.org/abs/1709.09890.