Deep Architectural Style Classification

Paavani Dua [1] **Alan Flores-Lopez** [2] **Alex Wade** [2]

Abstract

This report presents an end-to-end approach for architectural style classification of building façades with modern deep learning techniques. Our approach uses transfer learning to extract feature vectors from images of buildings and uses a neural network to map those features to class labels, achieving an overall test accuracy of 75.7%. Additionally, we present human error results, several weaknesses in an existing dataset for the task, and preliminary results from a follow-up approach that uses object detection to help alleviate the problem of mixed architectural styles.

1. Introduction

Ground-truth classification of architectural styles can be highly subjective. Architectural historians have considered architectural "style" from many different perspectives—the term can be used to group together buildings by particular visual traits or by cultural trends and building strategies that may be invisible on the surface (Hopkins, 2014). The existence of buildings which combine elements of multiple styles or defy easy classification further complicates the task.

To approach the task, we first assume the ground truth of an existing dataset (Xu et al., 2014). Specifically, we first seek to solve the problem defined by the creators of this dataset: classifying an image of a building façade into one of 26[1] architectural styles such as Byzantine, Romanesque, or Tudor Revival (see Figure 1). But as we will see, several problems with the original dataset and with the nature of the learning task hearken back to the complexities of architectural style classification. Concretely, besides presenting a model to solve the original task, we 1) discuss several weaknesses in the original dataset created by Xu et al. and 2) briefly propose and present preliminary results for an object-detection-based extension which addresses the problem of detecting multiple styles in the same building façade. To the best of our knowledge, this work is the first instance of modern deep learning techniques applied to this specific classification task.

The next section outlines related work. The following two sections present our end-to-end approach, results, analysis, and issues with the original dataset. The section after that presents an approach for finer-grained style classification. Finally, we conclude and suggest further work.

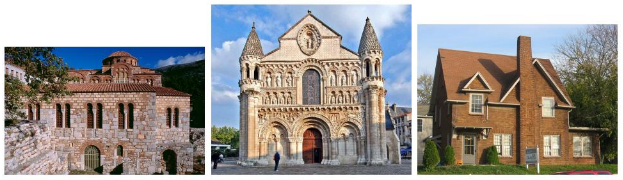

(a) Byzantine (b) Romanesque (c) Tudor Revival

Figure 1. **Architectural Styles.** Three sample building façades and their styles.

2. Related Work

Prior to 2016, work in the classification of architectural style or architectural elements was based on traditional machine learning approaches or on approaches with hand-selected features. Work by Shalunts focused on classifying various architectural elements into a small sets of classes using hand-crafted features and clustering (Shalunts et al., 2011; 2012a;b; Shalunts, 2015). Others took a graph-based approach to extract visual patterns for a similar problem in architectural image classification (Chu & Tsai, 2012) . Two other studies used modern machine learning techniques, although their approaches were not end-to-end and did not use ConvNets (Xu et al., 2014; Zhang et al., 2014).

Modern deep learning was first used in 2016 to classify a hand-crafted dataset of Mexican historical buildings into one of three classes (Obeso et al., 2016), creating models architectures from scratch and comparing their performance to AlexNet, although this work did not acknowledge the problem of eclecticism/mixed styles in classification. The

[1]Department of Electrical Engineering, Stanford University [2]Department of Computer Science, Stanford University. Correspondence to: Paavani Dua <paavanid@stanford.edu>, Alan Flores-Lopez <alanf94@stanford.edu>, Alex Wade <awade@stanford.edu>.

[1]The original dataset had 25 classes, but we added a 26th "no architecture' class.

following year, another work also applied modern deep learning and transfer learning techniques to classify architectural elements into their identity (window, pillar, etc.) (Llamas et al., 2017), and Chris Pesto trained three ConvNets to classify the architectural styles of images from Zillow for a CS231N final project (Pesto, 2017). In 2018, deep learning was applied to classify architectural elements by architect (Yoshimura et al., 2018).

Despite existing similar work, ours appears to be the first to apply modern deep learning techniques to the exact problem defined by Xu et al. in 2014, and one of the few to acknowledge the problem of mixed styles.

3. Methods

3.1. Data

We extended a publicly available dataset built by Xu et al. (2014) for our experiments. The original dataset had 25 classes of architectural styles such as Russian Revival, Postmodern, Bauhaus, and so on – it was gathered by recursively scraping Wikimedia's page on "Architecture by Style." We added 259 images of "no architectural style" to obtain a total of 5,053 images. See Table 1 for the count distribution among classes. All images (except the "no architecture" class) contain only building façades, and exclude any interior decorations or severe scale and orientation changes. All images are in RGB format. Image resolutions range from 300×200 to 3000×4000, with most images around a 800 $\times$ 600 resolution – all images are resized to 224×224 for training and evaluation.

Split. Because our dataset is not very large, we split it randomly into 70% train, 20% dev, and 10% test sets. We ensured that the class distributions remained the same in each of the train, dev, and test sets.

Data Augmentation. We augmented each original image in the dataset with horizontal flipping, rotations of -15, -5, 5, and 15 degrees, and four random crops. This increased the size of the original training set by 10x.

3.2. End-to-end classification

Figure 2 shows our final end-to-end architecture. We use a pretrained MobileNet V2 (Sandler et al., 2018) architecture trained on ImageNet (Deng et al., 2009) as an image feature extractor. The MobileNet V2 extractor takes images of size 224×224 and outputs 1792 features. We pass these 1792 features into two fully-connected layers of size 500 and 26, the first with a ReLU activation and the latter with Softmax. Figure 3 shows our choice of loss function.

We use an Adam optimizer with a learning rate of 0.001, a batch size of 256, L_1 regularization with $\lambda_1 = 1 \times 10^{-5}$, and dropout with a keep-probability of 95%. We ended up

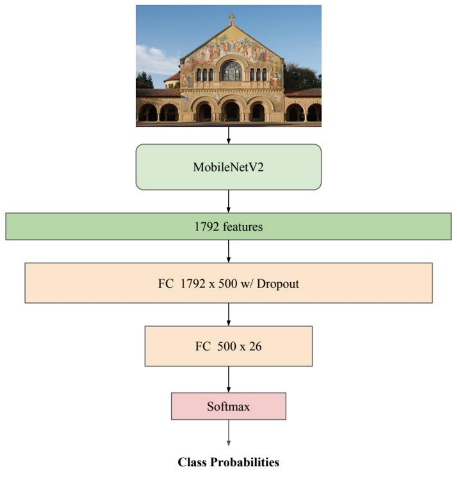


Figure 2. **End-to-end architecture.** We use a MobileNet V2 architecture trained on ImageNet to extract features from images of building façades and a neural network to map those features to a probability distribution over style class labels.

$$\text{Cross-Entropy Loss} = \frac{1}{|D|} \sum_{(x,y)\in D} \Big[-f_y + \log \sum_j e^{f_j} \Big]$$

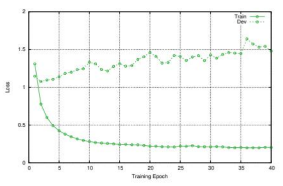
$$\text{Total Loss} = \text{Cross-Entropy Loss} + \lambda_1 L_1(W) + \lambda_2 L_2(W)$$

Figure 3. **Loss function.** We minimize the cross-entropy loss as it is standard for softmax outputs – we explored using L_1 and L_2 regularization to help mitigate overfitting. Here D is our dataset, x is an example image, y is its style label, f_j is the value of the j-th entry of the class scores vector for the current training example x, and W is the set of weights in the post-feature extraction neural network. We do not backpropagate into the weights of the pretrained architecture.

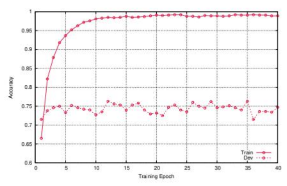
setting our L_2 parameter to zero, but did explore its use. We use TensorFlow (Abadi et al., 2015) for all our experiments.

3.3. Hyperparameter Tuning and Architecture Search

The final architecture described in the previous section was the result of a thorough hyperparameter and architecture search. We tested five different pretrained image feature extractors with our task, including a ResNet variant (He et al., 2016), Inception V3 (Szegedy et al., 2015), and a large NASNet (Zoph et al., 2017) architecture, but it was MobileNet V2 that achieved the highest dev accuracy on a preliminary experiment (75%, compared to about 70%). This result, together with fast training times and the idea that an architectural classifier could be useful on mobile devices, made MobileNet V2 the go-to architecture for the rest of our experiments.



(a) Loss.



(b) Accuracy

Figure 4. **Loss and accuracy curves**. The loss and accuracy curves for the final model – dev set accuracy increases rapidly in one epoch, but then tapers off, remaining below train set accuracy. Despite experimenting with L_1, L_2, and dropout regularization, as well as data augmentation, we could not improve this variance with the existing dataset.

Having fixed MobileNet V2 as our feature extractor, we used a random search to tune the learning rate, batch size, number of fully-connected layers (1 to 4) in our output neural network, the coefficients of L_1 and L_2 regularization, and whether or not to use batch norm and what momentum to use with it. Lastly, we took the best sets of parameters from this experiment to tune the keep probability of dropout between fully-connected layers, which gave us our final architecture[2].

The vast majority of our experiments (even those with the highest dev accuracy) overfit the training set to about 99.9% accuracy. This meant that the feature extractor gave rich enough features for our algorithms to overfit the training images. For this reason, we did not experiment with fine-tuning the pretrained architectures and instead focused our efforts on regularization and avoiding variance.

For comparison, we include results from a baseline CNN model in the results section.

4. Results

Table 1 displays per-class test set performance metrics from our own baseline CNN and our final models. The corresponding train and dev accuracy and loss curves are in Figure 4.

Xu et al. reported an accuracy of 46.2% with their multinomial latent logistic regression approach (per-class accuracy was not reported) (Xu et al., 2014). In comparison, we achieved 55.4% accuracy on a baseline CNN architecture[3], and 75.7% accuracy using our MobileNetV2 transfer model.

[2]We ran tuning experiments for about 5 days on an AWS p2.xlarge GPU instance.

[3]Our baseline consists of a series of 3x3 convolutional layers with batch normalization, ReLU activations, and 2x2 max pool layers. Two fully-connected layers map the final volumes into the 26 architecture styles.

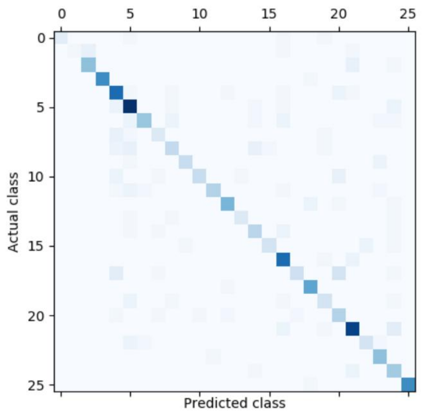


Figure 5. **Test set confusion matrix.** MobileNetV2 transfer learning. A few classes such as American Craftsman (class 1) have a significant number of images classified erroneously. Most of the American Craftsman images were classified as American Foursquare (class 2), which is visually very similar. Similarly, the International (class 17) style is often misclassified as Art Deco (class 4) or Postmodern (class 20).

Figure 5 presents a confusion matrix of the *test set* results from our architecture. While our classifications are usually correct, there are a few systematic errors apparent from the diagram, described in the caption.

Human performance. We consulted with a trained architect[4] to approximate the Bayes error rate on the architectural classification task. We presented her with all the labeled training data (not augmented), as well as a set of 25 images randomly chosen from the dev set with the labels removed. All images were reduced to the resolution we gave the classifier. We asked her to classify the 25 images into styles based on her architectural knowledge and the example data. Her accuracy on the 25 samples was 56%. Many of her misclassifications were similar to our model's, with visually similar classes mistaken for each other, like International and Postmodern.

4.1. Discussion

Dataset and Task Limitations. The architect we consulted with also noticed a number of images which she believed were mislabeled, as well as systematic misclassifications. Given that the original labels were added by Wikimedia users, this is indeed a valid concern with the data quality.

In addition, she discussed issues with the task itself. Many of her mislabeled images were of American architecture, which, especially in the late 19th century, consisted of a

[4]Thanks to C.C. Ying.

		Baseline CNN			Final Architecture		
	Count	F1	Precision	Recall	F1	Precision	Recall
Achaemenid	69	61.5	66.7	57.1	72.7	100.0	57.1
American Foursquare	59	**25.0**	50.0	16.7	**28.6**	100.0	16.7
American Craftsman	195	52.9	64.3	45.0	82.1	84.2	80.0
Ancient Egyptian	256	98.0	100.0	96.2	98.0	100.0	96.1
Art Deco	366	47.1	41.7	54.1	73.1	66.7	81.1
Art Nouveau	450	60.5	65.0	56.5	76.4	69.6	84.8
Baroque	239	57.8	61.9	54.2	73.1	88.2	62.5
Bauhaus	92	**28.6**	50.0	20.0	**52.6**	55.6	50.0
Beaux-Arts	191	50.0	56.3	45.0	**55.6**	62.5	50.0
Byzantine	111	**20.0**	25.0	16.7	81.8	90	75.0
Chicago School	153	51.4	47.4	56.3	69.2	90	56.3
Colonial	177	**37.2**	32.0	44.4	77.4	92.3	66.7
Deconstructivism	213	50.0	41.2	63.6	83.7	85.7	81.8
Edwardian	79	43.5	33.3	62.5	71.4	83.3	62.5
Georgian	154	47.1	44.4	50.0	66.7	64.7	68.8
Gothic	109	55.6	71.4	45.5	73.7	87.5	63.6
Greek Revival	327	65.7	64.7	66.7	80.0	71.4	90.9
International	207	53.7	55.0	52.4	**53.3**	88.9	38.1
Novelty	212	51.3	58.8	45.5	89.4	84.0	95.4
Palladian	113	43.5	45.5	41.7	58.3	58.3	58.3
Postmodern	163	44.4	60.0	35.3	**57.1**	48.0	70.6
Queen Anne	425	64.8	53.8	81.4	81.8	80.0	83.7
Romanesque	107	50.0	55.6	45.5	70.0	77.8	63.6
Russian Revival	165	48.6	45.0	52.9	84.2	76.2	94.1
Tudor Revival	162	**29.6**	40.0	23.5	62.2	50.0	82.4
No Architecture	212	87.0	100.0	76.9	98.0	100.0	96.1
Overall	5,053	55.1	57.4	55.4	75.3	78.3	75.7

Table 1. **Results.** Per-class counts and result metrics for our baseline and final model – we present standard F1, Recall, and Precision metrics. The best and worst few F1 scores for each model are underlined and bolded, respectively. Globally, Xu et al. (2014) reported 46.21% accuracy on a test set of 1,716 images across the 25 original architecture styles. Our baseline CNN's global accuracy was 55.4%, and our best approach's accuracy was 75.7%.

mix of styles which elude easy classification into one category. She described one building in particular as a "bizarre" amalgam of styles (Figure 6).

Indeed, our model reflects these two problems. The test set confusion matrix reveals that American Foursquare buildings were more often classified as American Craftsman than not. One possibility is that many of the American Foursquare images are actually mislabeled and could more canonically be considered American Craftsman buildings. Another possibility is that some of these buildings embody elements of both styles, and that due to the greater number of American Craftsman examples in the training set, these mixed buildings default to American Craftsman. One example of this phenomemon is shown in Figure 7 along with a true American Craftsman house. Additionally, we observe that a few images labeled as Art Deco and Art Nouveau are classified as each other, corroborating her observation that Art Deco and Art Nouveau images often seem mistaken for each other in the dataset.

Ultimately, our classifier's accuracy appears high given the limitations of the dataset as reflected in human error, and many of our classifier's mistakes may be explained away by data irregularities or inherent task difficulties.

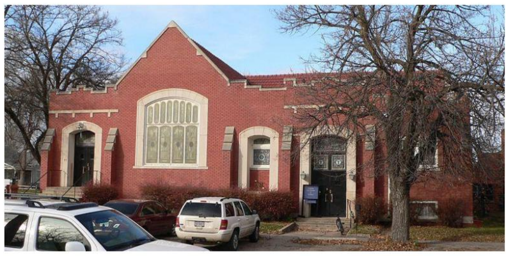
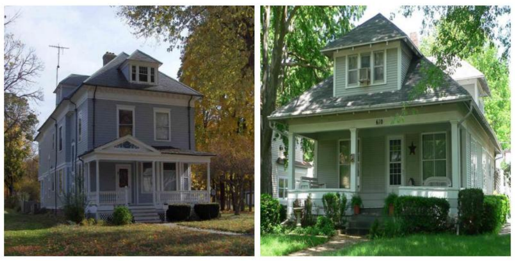


Figure 6. **Eclectic Architecture.** An example of a "bizarre" building from the training set, with a mix of architectural styles. The dataset labeled it as Tudor Revival, with which our expert disagreed.

5. Fine-Grained Classification

To begin to address the problem of buildings with mixed styles, we implemented an approach that leverages object de-



(a) American Foursquare (b) American Craftsman

Figure 7. **American Craftsman and American Foursquare.** On left, a supposedly American Foursquare building misclassified as American Craftsman. On right, a correctly classified American Craftsman house. Our model had difficulties distinguishing between the two styles, and our human expert expressed concerns that some exemplars of each in the dataset were not sufficiently unique to the labeled style.

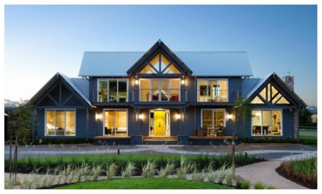
tection as an intermediate step in the classification pipeline. While a traditional deep learning approach gives a single label to an entire image without explanation and without finer-grained analysis, our intuition was that breaking the image down into its components and classifying those could yield a more nuanced analysis of building style.

Generating elements dataset. First, we took the original architectural style classification dataset, and used an object detection architecture (Huang et al., 2017) trained on the Open Images (Krasin et al., 2017) dataset to extract objects from each image. We extracted objects from a list of classes relevant to architectural classification (e.g. Sculpture, Door, Window, Tower, Stairs, etc.) and only took objects classified with higher than 50% confidence (see Figure 8). This process resulted in a new "elements" dataset with 4,822, 1,401, and 712 examples in the train, dev, and test set respectively.

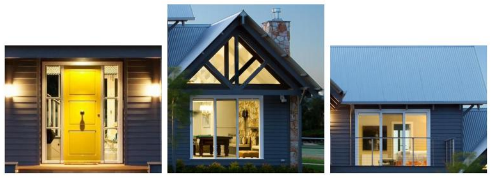
Experiments with elements dataset We used the same architecture and a similar training regime as for the original approach to learn a classifier for architectural *elements*. This achieved an 66.9% test accuracy on this new dataset. Figure 8 shows an example style classification output of the pipeline.

6. Conclusion

In this work we presented an end-to-end architectural style classification model that achieves 75.7% test accuracy, outperforming a human expert (56.0%), a baseline CNN (55.4%), and previous work (46.2%). At the same time, we illuminated some inherent shortcomings with the problem itself: not only may our existing dataset be inaccurately labeled, but the task itself is often intractable, as buildings are not always neatly classifiable into a single architectural style. Crafting an expert-curated dataset with annotated



(a) Original



(b) Door (c) 'House' (d) Window

Figure 8. **Sample output from elements pipeline**. We extracted architecture-related objects from images automatically to generate an elements architectural style dataset. In this eclectic example, the overall image was classified as Queen Anne, while the door was classified as Georgian, the 'house' Novelty, and the window American Craftsman.

building *features* would address both of these outstanding issues and is the logical next step for future work.

Lastly, as a prelude to these future directions, we present early results from an object-detection-based classifier to provide finer-grained architectural style classifications. Such a nuanced approach to architectural style classification is more akin to how architects themselves would likely approach the task, and appears to be a promising avenue for future work.

7. Code and Contributions

7.1. Code

Our code can be found in this public GitHub repo: `https://github.com/alanefl/archclass/`.

7.2. Contributions

Paavani performed and wrote about data augmentation, found and added the 26th 'no architecture' images to the dataset, and designed the final poster.

Alan wrote the scripts to fetch and prepare the dataset for training, coded the architectures, ran all hyperparameter tuning experiments, and built the object detection extension. Alan also prepared most of the report except results and the conclusion.

Alex wrote code for both global and per-class metric measurement as well as confusion. Alex also refactored some of the training and model code. Alex's code also utilized TensorBoard to display sample misclassified images for error analysis. Alex prepared the human version of the classification task, interviewed the architect, and wrote up the results and conclusion of this report.

References

Zillow. https://www.zillow.com/.

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL http://tensorflow.org/. Software available from tensorflow.org.

Chu, W.-T. and Tsai, M.-H. Visual pattern discovery for architecture image classification and product image search. In *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, pp. 27. ACM, 2012.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016. URL http://arxiv.org/abs/1603.05027.

Hopkins, O. *Architectural Styles : A Visual Guide*. Laurence King Publishing, 2014. ISBN 9781780671635. URL https://stanford.idm.oclc.org/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=926206&site=ehost-live&scope=site.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7310–7311, 2017.

Krasin, I., Duerig, T., Alldrin, N., Ferrari, V., Abu-El-Haija, S., Kuznetsova, A., Rom, H., Uijlings, J., Popov, S., Kamali, S., Malloci, M., Pont-Tuset, J., Veit, A., Belongie, S., Gomes, V., Gupta, A., Sun, C., Chechik, G., Cai, D., Feng, Z., Narayanan, D., and Murphy, K. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://storage.googleapis.com/openimages/web/index.html*, 2017.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Llamas, J., M Lerones, P., Medina, R., Zalama, E., and Gómez-García-Bermejo, J. Classification of architectural heritage images using deep learning techniques. *Applied Sciences*, 7(10):992, 2017.

Obeso, A. M., Benois-Pineau, J., Acosta, A. Á. R., and Vázquez, M. S. G. Architectural style classification of mexican historical buildings using deep convolutional neural networks and sparse features. *Journal of Electronic Imaging*, 26(1):011016, 2016.

Pesto, C. Classifying u.s. houses by architectural style using convolutional neural networks. Not published, Stanford CS 231N final report, 2017. URL http://cs231n.stanford.edu/reports/2017/pdfs/126.pdf.

Sandler, M. B., Howard, A. G., Zhu, M., Zhmoginov, A., and Chen, L.-C. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, abs/1801.04381, 2018.

Shalunts, G. Architectural style classification of building facade towers. In *International Symposium on Visual Computing*, pp. 285–294. Springer, 2015.

Shalunts, G., Haxhimusa, Y., and Sablatnig, R. Architectural style classification of building facade windows. In *International Symposium on Visual Computing*, pp. 280–289. Springer, 2011.

Shalunts, G., Haxhimusa, Y., and Sablatnig, R. Architectural style classification of domes. In *International Symposium on Visual Computing*, pp. 420–429. Springer, 2012a.

Shalunts, G., Haxhimusa, Y., and Sablatnig, R. Classification of gothic and baroque architectural elements. In *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 316–319. IEEE, 2012b.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. URL http://arxiv.org/abs/1512.00567.

Xu, Z., Tao, D., Zhang, Y., Wu, J., and Tsoi, A. C. Architectural style classification using multinomial latent logistic regression. In *European Conference on Computer Vision*, pp. 600–615. Springer, 2014.

Yoshimura, Y., Cai, B. Y., Wang, Z., and Ratti, C. Deep learning architect: Classification for architectural design through the eye of artificial intelligence. *CoRR*, abs/1812.01714, 2018. URL `http://arxiv.org/abs/1812.01714`.

Zhang, L., Song, M., Liu, X., Sun, L., Chen, C., and Bu, J. Recognizing architecture styles by hierarchical sparse coding of blocklets. *Information Sciences*, 254:141–154, 2014.

Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. V. Learning transferable architectures for scalable image recognition. *CoRR*, abs/1707.07012, 2017. URL `http://arxiv.org/abs/1707.07012`.