Extracting Indexing Features for CBR from Deep Neural Networks: A Transfer Learning Approach

Zachary Wilkerson, David Leake[0000−0002−8666−3416], Vibhas Vats, and David Crandall

Luddy School of Informatics, Computing, and Engineering, Indiana University
Bloomington IN 47408, USA
{zachwilk,leake,vkvats,djcran}@indiana.edu

Abstract. High-quality indices are essential for accurate retrieval in case-based reasoning. However, in some domains, indexing knowledge may be incomplete, unavailable, or unfeasible to obtain by knowledge acquisition, making knowledge-light machine learning methods an appealing alternative for generating indexing features. In response, previous work has developed promising methods for extracting indexing features from deep neural networks trained on case data. However, it has also underlined that CBR using features extracted from a deep neural network achieves low accuracy in domains for which the network itself has low accuracy when trained from scratch. This is a special concern for CBR feature extraction because the ability of CBR to reason successfully in "small-data" domains has been seen as one of its benefits. This paper reports on work investigating the hypothesis that transfer learning may help decrease the data requirements for index extraction. Specifically, it examines how model pretraining affects the quality of extracted indexing features for case-based classification, measured by the performance of a case-based classifier using those features for retrieval. Experimental results suggest that using a pretrained deep learning model for feature extraction can improve classification accuracy and consistency compared to using similar models trained from scratch. An unexpected and intriguing result is that the case-based classifier using extracted features outperformed analogous deep learning classifiers for the tested dataset.

Keywords: Case-Based Reasoning, Deep Learning, Feature Learning, Hybrid Systems, Indexing, Integrated Systems, Retrieval, Transfer Learning

1 Introduction

The performance of case-based reasoning (CBR) systems depends critically on retrieving useful cases from the case base. This retrieval is generally based on a feature vocabulary used to index the cases in the case base. Traditionally, such a vocabulary is developed through knowledge engineering (e.g., [14, 23, 35]), with indices assigned to new problems by situation assessment. This knowledge-based approach can be effective in identifying key features and facilitating human-understandable explanations for retrievals. However, knowledge engineering is costly and generating knowledge-engineered feature sets may be unfeasible for some domains, such as image processing.

Deep learning (DL) classification methods avoid traditional knowledge acquisition and provide high accuracy across a variety of domains, including image classification [10]. This stems from the ability of DL models to learn useful feature information from raw data. Consequently, the application of deep learning methods to learning features for case retrieval has received attention from the CBR community (e.g., [3, 33, 37, 38]). DL feature generation for case-based classifiers has three potential benefits: to increase accuracy, to decrease the knowledge engineering burden, and to enable case-based classification in domains for which retrieval features are unfeasible to generate by hand (e.g., image classification). Compared to using a pure DL classifier, the hybrid approach provides an interpretability benefit: The hybrid system can explain its decisions by presenting the retrieved cases for humans to assess their relevance—which can be an effective explanation strategy even when retrieval features are unexplained [16].

Many factors may affect the quality of case retrieval features extracted from a DL model. We have explored how the number of features extracted, network depth at which features are extracted, and extractor model architecture influence feature quality [26, 27]. We have also explored how using existing knowledge-engineered and extracted features together improves classification accuracy overall [40]. These experiments provided useful information about how to design feature extraction systems, but the experimental results also pointed to a concern. One of the motivations for using CBR is to enable effective reasoning in "small-data" domains for which only limited numbers of cases may be available or necessary [24]. Consequently, we tested the methods on DL models trained from limited training data to examine performance for small-data domains. Our results showed that DL models trained from scratch on limited training data overfitted or failed to converge, resulting in poor-quality features and suboptimal CBR performance. Thus, in such domains, lack of data limits the effectiveness of feature extraction from DL models. This is problematic for the goal of leveraging DL while retaining the ability of CBR to reason successfully with limited data.

Extensive DL research has shown the benefit of transfer learning, in which DL models thoroughly trained for one domain are leveraged and specialized to new related tasks. This led us to consider whether applying transfer learning to a DL network used for feature extraction could improve the quality of case retrieval features produced with limited domain data. This paper presents experimental results on the use of transfer learning for DL feature extraction, examining the performance achieved for various DL architectures.

The paper begins with an overview of our general DL-CBR feature extraction approach and summary of our previous studies, which provide a baseline for the current work. We then present an evaluation of several DL feature extraction models using networks that have been pretrained and then specialized via training on a novel test domain. Our earlier evaluation of these architectures using training from scratch suggested that more complex DL models (e.g., Inception V3 and DenseNet121) are potentially more prone to overfitting, producing lower-quality features. In contrast, our new experimental results show that using pretrained versions of these same DL models can produce high-quality features, providing high classification accuracy for the CBR system. Surprisingly, the overall DL-CBR system accuracy rivals and occasionally surpasses the classification accuracy of using the pretrained DL architecture end-to-end in our experiments. This potential performance benefit is intriguing, and we discuss potential explanations in Section 5.3. The DL-CBR system also appears to be more consistent on average, yielding lower average standard deviation over all tests. Especially when combined with the interpretability of the DL-CBR system, these results appear very promising.

2 Related Work

Integrations of DL with CBR are appealing for goals such as explaining DL and increasing CBR performance. For example, similarity metric learners that focus on class-wise comparison of examples, such as Siamese networks [22], relation networks [36], and matching networks [34], can be used for similarity assessment in CBR (e.g., [4, 28, 30]), and network architectures can be designed for features to be compared against prototypes to guide the model decision [7, 11, 29]. Other research has investigated DL-CBR integration to explain DL models, as with *post-hoc* feature-level explanations [5] and "twin systems" [19]. Finally, and of particular interest to this paper, previous research has explored ways that DL systems may be leveraged to extract feature information for CBR systems, as described in Section 2.1.

2.1 Extracting DL Features for Case Retrieval

Case retrieval features have traditionally been developed by knowledge engineering [14, 23, 35]. However, acquiring expert-based feature sets may be unfeasible for some domains, and knowledge engineering can be costly. Symbolic learning methods have successfully been applied to this problem using existing feature vocabularies (e.g., [6, 8, 9, 12, 15]), but DL feature extraction is appealing to enable the system to develop its own feature vocabulary (e.g., [3, 33, 38]) and to develop features for domains such as image classification. For example, Turner et al. use a CBR system to perform relative classification on examples for which the DL system lacks confidence in its decision (esp. for examples from novel classes) [37, 38]. In this way, the CBR system leverages features extracted from the DL system and clusters examples to form implicit classes. Sani et al. use a similar approach for feature extraction but always use the CBR system to render model decisions [33]. They highlight that the ability of the hybrid system to explain its decision through case presentation, combined with its accurate performance, make it a promising model for the types of domains in their case study.

These approaches make assumptions that are challenged in our previous work. First, they extract feature vectors from between the convolution and densely-connected layers in the DL model; we found that better quality features may be extracted from between the densely-connected layers and the output layer, where (ideally) features have been combined into more complex indices by the network's densely-connected layers (see Section 3.2 for details) [26]. Second, previous work assumes that the DL system is the sole source of features for case retrieval. However, knowledge-engineered features may be available for some domains, even if those features alone are insufficient. We showed that in some cases using a combination of expert-generated and DL-derived features can significantly benefit system performance [27, 40]. In addition, we showed sensitivity of feature quality to several model-level parameterizations, such as number of features extracted [26] and DL model architecture [27].

2.2 Transfer Learning

Transfer learning exploits the results of learning for one task to improve performance on another similar task [42]. CBR itself has been advocated as a transfer learning method [21], and knowledge transfer to improve CBR has been studied in contexts such as cross-case-base adaptation [25] and index revision during long-term system operation [17, 31].

In deep learning, the need for large datasets and extensive training to achieve strong performance has led to great interest in exploiting prior models to improve training in new domains. In this approach, weights from an already-trained DL model are used to initialize the weights for a new model, which is then specialized on a different dataset during training. Transfer learning by using pretrained models has proven powerful for overcoming issues arising from limited data when training end-to-end DL systems [32]. Because our previous studies of DL feature extraction showed the difficulty of training the networks for small-data domains, we hypothesized that improving the DL model by using pretrained models would improve the quality of extracted features as well.

3 Our Approach to DL Feature Extraction for Case-based Classification

3.1 Convolutional Neural Network Structure

As context for describing our method, we briefly summarize neural network structure. Neural network models can be broadly conceptualized as a system of layers that use features from previous layers to inform feature combinations that occur in subsequent layers. Taken together, these layers process raw input data into low-level atomic features that are then combined into more complex mid- and high-level features that are used to render the final model decision. In this high-level abstraction of feature learning, successive sub-sampling and feature combination are especially fundamental to convolutional neural network (CNN) models, which are the primary models used in our research.

CNNs extract features by using convolution layers that employ sliding matrix operations to condense multi-dimensional raw input data into numeric features:

$$O_{xy} = \sum_{i=-k}^{k} \sum_{j=-l}^{l} F_{ij}(I_{(x-i)(y-j)}) \tag{1}$$

That is, by applying a convolution filter F of size $(2k+1) \times (2l+1)$ to the input data I, the corresponding output feature O_{xy} is the inner product of the filter, and the region of the input centered on (x, y) and defined by the dimensions of the filter. Thus, successive convolutions map the feature information from the previous layer into increasingly refined feature sets that ideally represent key elements of the original input (e.g., patterns, and shapes for image data). Successive convolutions are flattened into feature vectors and used as input to densely-connected (multi-layer perceptron) layers to generate more complex features. The outputs of these combinations inform the final model decision.

3.2 Extracting Network Features for Case Retrieval

Many CBR systems use feature-vector problem representations. Consequently, the feature vectors generated from convolution in CNNs map naturally to case indices, and post-convolution features have proven useful for case retrieval in CBR [33, 37, 38]. However, we have found that extracting features after the feature combination step before the output layer can provide higher quality features—that is, features that result in higher classification accuracy [26]. This extraction approach has two potential benefits in addition to increased accuracy: First, the densely-connected layers deeper in the network conveniently may be parameterized to reduce the size of the feature set to mitigate "curse of dimensionality" effects while minimizing side-effects affecting the model's ability to converge, and second, extracting post-combination may be more straightforward than extracting at a shallower location for more complex DL models, whose layers are often more interconnected than more "linear" models such as AlexNet and VGG [20].

Figure 1 illustrates our basic approach to feature extraction [27]. It applies post-combination feature extraction and allows for using extracted features in concert with existing knowledge-engineered features, if they are available.

3.3 Lessons from Our Previous Studies

Building on our basic approach for DL feature extraction for case retrieval, we have explored variants to improve this approach, in the context of image classification tasks. Our work resulted in the following observations:

1. **The number of features extracted significantly impacts classification accuracy.** Comparing high-dimensional feature vectors for CBR similarity calculations can result in a "curse of dimensionality," where individual features contribute minimally to the overall similarity calculation. Consequently, there exists a point at which increasing the number of features harms

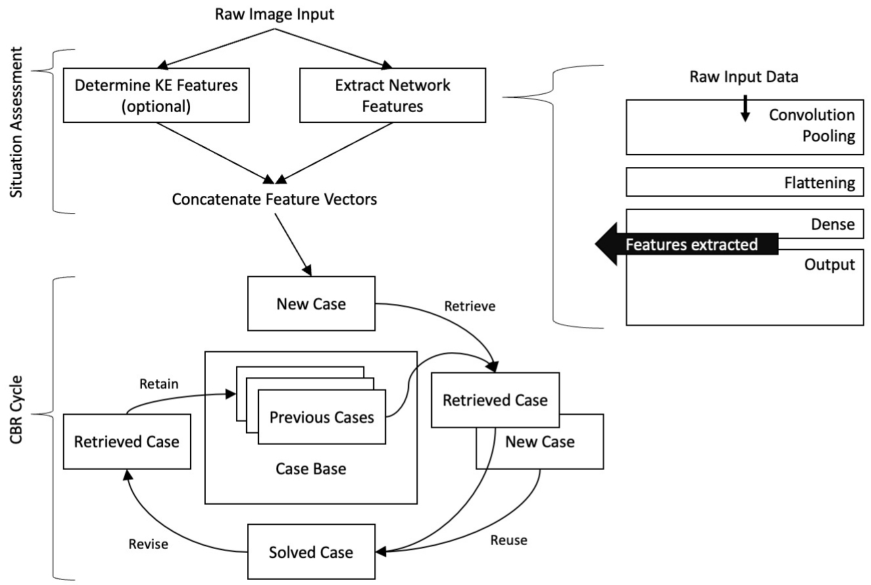


Fig. 1: Features are extracted from the DL model (right), combined with knowledge-engineered features if applicable (top left), and then used to inform case retrieval (figure from Leake et al. [27]; CBR cycle after Aamodt and Plaza [1]). We illustrate CNNs for feature extraction, but this approach may be generalized to other DL models.

case retrieval performance. Conversely, as the number of features extracted decreases, the DL model's representational power also decreases, to the point of failing to converge on a representative feature set, harming classification accuracy as well. Thus, there exists a "sweet spot" for which the number of features extracted is large enough to ensure convergence, but small enough to avoid a curse of dimensionality [26].

2. **Using existing knowledge-engineered features along with extracted features can increase classification accuracy.** When available, knowledge-engineered features can supplement features extracted from the DL system to enable higher classification accuracy [40]. In addition, the inclusion of useful knowledge-engineered features can offset the negative effects of poor model training [27]. However, in both of these cases, overall performance depends on the reliability of the knowledge-engineered features; features whose values are noisy or that do not reliably characterize the case base with respect to the task at hand may have a muted or even harmful impact on model performance.
3. **Small-data domains are challenging for feature extraction, and DL model architecture influences classification accuracy.** Our previous experimental results showed that DL-CBR feature extraction systems

trained from scratch on small-data domains exhibit suboptimal classification accuracy, limiting their applicability to such domains. They also challenged our hypothesis that more recently-developed complex DL models would produce higher-quality features; the deepest model tested (DenseNet121) did indeed have the best relative performance, but it only marginally outperformed the shallowest model tested (AlexNet). These conclusions motivated us to investigate transfer learning to increase robustness for small-data domains, as well as the opportunity for transfer learning to impact the relative performance of different DL feature extractor models.

4 Evaluation

4.1 Hypotheses

We performed computer experiments to evaluate the general hypothesis that pretrained models could result in higher quality feature extraction for small-data domains. Specifically, for a selection of the neural architectures tested in our previous feature extraction research, we investigate the following hypotheses:

H1: **Using transfer learning for feature extraction will produce better quality features than training from scratch.** This will manifest in higher classification accuracy on both train and test data.

H2: **With transfer learning, feature extraction using more complex models (e.g., Inception and DenseNet) will lead to higher classification accuracy.** This hypothesized behavior contrasts with our previous findings without transfer learning [27]. We hypothesize the previous detriment from these models stems predominantly from training from scratch on a small dataset and that transfer learning will address this.

H3: **Using a case-based classifier will lead to lower classification accuracy than using an end-to-end DL system.** DL classifiers can achieve high accuracy, which may be increased through transfer learning. Additionally, our experimental systems are knowledge-light, preventing them from exploiting domain knowledge that can strengthen CBR performance. However, as our method leverages DL to extract case retrieval features, we expect that any accuracy difference between our model and an end-to-end DL system will be sufficiently small that in some domains it may be justified by the benefit of CBR interpretability.

4.2 Evaluation Strategy and Testbed System

Building on our previous work training DL feature extraction models from scratch, we tested three of those DL models that have readily-available pretrained versions through the Tensorflow Applications module [2] (i.e., VGG-19, Inception V3, and DenseNet121 [20]). In contrast with our previous work, we do not consider using supplementary knowledge-engineered features for retrieval, and we modify densely-connected layer structures to keep the number of extracted features constant across all architectures. Furthermore, since this research focuses specifically on retrieval, our CBR system has no adaptation component. It performs classification using a one-nearest-neighbor approach based on Manhattan distance. All extracted features were unweighted, but in principle weights could be learned [39].

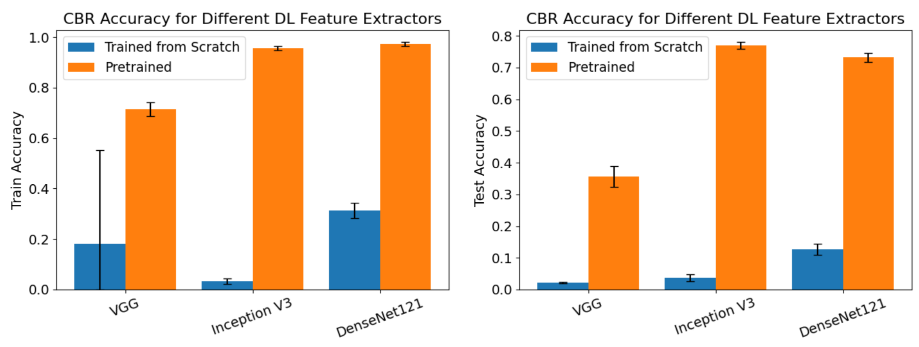
We compare the accuracy of the case-based classifiers to a DL classifier baseline. For that baseline, we use the end-to-end classification accuracy of the DL network from which features are extracted for the case-based classifier (i.e., using the DL system's output, rather than extracting features and performing classification with the CBR system). Features are extracted following the densely-connected layers as in our previous work [26].

4.3 Dataset and Experimental Parameters

Each of the models used for feature extraction is pretrained on ImageNet [13] using the built-in functionality provided in the Tensorflow Applications library. Each pretrained model is imported without its top (i.e., the densely-connected layers), which is appended manually so that 1024 features are extracted from every model. Then, each model is further specialized by training on examples from the Animals with Attributes 2 (AwA2) dataset [41] for a maximum of 50 epochs; early stopping with a patience value of two epochs (i.e., halting training if validation accuracy does not improve for two consecutive epochs) is used to minimize overfitting. The number of training examples used for specialization is varied from 1024 to 8192 examples in increments of 1024, and additional experiments with 512 training examples are conducted to enable direct comparison to our previous results. Training examples are selected randomly from the larger dataset. Testing is conducted for both DL and case-based classifiers on both the training set—to estimate an upper bound on classification accuracy—and on an independently-selected test set of the same size and composition to estimate a general classification accuracy. All trials are conducted thirty times to establish reliable mean and standard deviation accuracy values.

5 Results and Discussion

Figure 2 compares our previous experimental results for training from scratch and our results with transfer learning for a similar number of features, and Figures 3 and 4 show our results using transfer learning for all tested numbers of extracted features. Several broad trends are apparent. First, and most evident, using transfer learning to pretrain DL feature extractors leads to substantially higher classification accuracy across the board than the same models trained from scratch; this supports H1. As the number of extracted features increases, the accuracy advantage of using pretrained models becomes particularly pronounced in some cases, enabling the DL-CBR model accuracy to surpass the accuracy of the DL classifier baseline. This exceeds the expectations set in H3, which predicted that the DL system would maintain superior classification accuracy,



(a) Evaluated on training data (b) Evaluated on testing data

Fig. 2: AwA2 classification accuracy on train (a) and test (b) data for the DL feature extractor architectures (using the corresponding 512 training example accuracy values from our new data), comparing evaluation on the AwA2 dataset using DL models trained from scratch on AwA2 (as in Leake et al. [27]) and using transfer learning (pretrained on ImageNet and specialized on AwA2, from these experiments). Error bars represent one standard deviation.

potentially with a limited difference. Using DenseNet for feature extraction led to the highest classification accuracy (Figures 3 and 4). As DenseNet is the most recent/advanced of the DL architectures explored in this work, this is consistent with H2; however, study of other other architectures is needed (e.g., larger DenseNet models, ResNet).

5.1 Training from Scratch versus Transfer Learning

In our previous work, we concluded that the impact of the DL architecture on classification accuracy was overshadowed by the impact of training set size [27]. That is, models across the board overfitted to training data or failed to converge due to the small training set size; some models (e.g., AlexNet and DenseNet) were somewhat resilient to this, but any differences were not dramatic. However, when models are pretrained and specialized on the same limited training data, the results are substantially different (Figure 2).

At the outset, even using a training set size comparable with the one used in Leake et al. [27], the DL-classification accuracy is significantly higher. This result is especially pronounced when using Inception or DenseNet architectures, both of which significantly outperform VGG-based feature extraction models. Furthermore, while there is an expected decrease between classification accuracy on the training set and the independent test set, the difference is generally more consistent across all models and is smaller in scale (around 30% or less). Our earlier results show that using a case-based classifier does not substantially offset the negative effects of training the DL model from scratch on limited data.

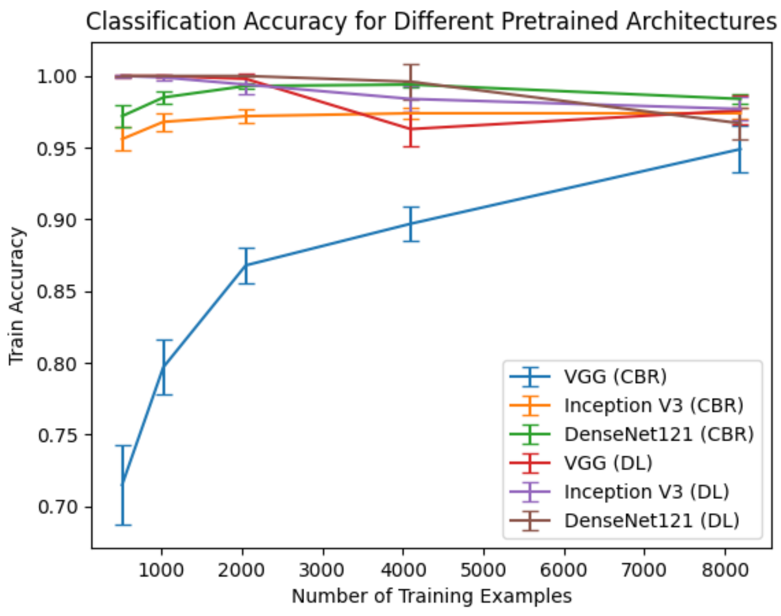


Fig. 3: Accuracy values for tested architectures, evaluated on the training set. Each DL architecture is evaluated both as a feature extractor for a case-based classifier (CBR) or as an end-to-end DL classifier (DL). Error bars represent one standard deviation.

Intuitively, such a model may be improved simply by considering more training data and/or training for more epochs. However, our results support transfer learning as an effective means to apply our DL feature extraction approach in data-sparse domains without requiring additional data or training cost.

5.2 Architectural Influences on Feature Quality

When using pretrained models, the choice of architecture significantly impacts extracted feature quality, as evidenced by significantly different classification accuracy values from the experiments. It is immediately apparent that in this context, shallower models such as VGG underperform significantly and are more dependent on having a "critical mass" of training data to achieve reasonable classification accuracy. Presumably, these conclusions may be extrapolated to related models such as AlexNet. Beyond this, the difference between DenseNet and Inception is significantly smaller, but it appears that DenseNet consistently performs better as a feature extraction architecture (Figures 3 and 4).

Based on these results, it is interesting to consider the factors that might account for DenseNet emerging as the best architecture among those tested in this

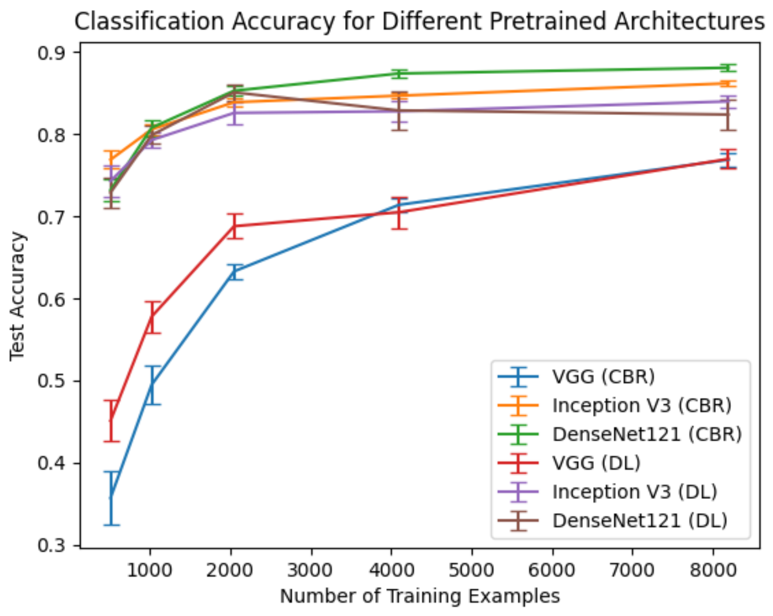


Fig. 4: Accuracy values for tested architectures, evaluated on an independently-selected test set. Each DL architecture is evaluated both as a feature extractor for a case-based classifier (CBR) or as an end-to-end DL classifier (DL). Error bars represent one standard deviation.

case study. The answer could be as simple as pointing to the same factors that make DenseNet preferred as an end-to-end DL architecture (i.e., DenseNet's greater depth). However, this is potentially less likely due to the comparable performance of the end-to-end models in our experiments (though it is possible that the CBR system is more strongly influenced than the end-to-end performances). It is also reasonable to consider whether the depth-wise parallelism of features afforded via DenseNet's "skip connections" [18] is more useful for CBR feature extraction than the lateral parallelism contained in Inception's inception modules. Further architecture testing (e.g., with ResNet, which also has skip connections) might help isolate any general structural influences.

5.3 Accuracy and Consistency Benefits of Case-Based Classification with DL Features vs. Pure DL

The most unexpected finding in these experiments is that DL-CBR classifiers may perform more accurately than the corresponding end-to-end DL system. In addition, the hybrid models were frequently more consistent, exhibiting a smaller average standard deviation (Table 1). This manifests most dramatically

Standard Deviation on Training Set

Model	Classifier	Avg. St. Dev.	# of Training Examples 512	1024	2048	4096	8192
VGG	DL	***0.0056***	0.001	0.001	0.004	0.012	0.010
	CBR	0.0174	0.028	0.019	0.012	0.012	0.016
Inception V3	DL	***0.0054***	0.001	0.002	0.007	0.009	0.008
	CBR	***0.0054***	0.008	0.006	0.005	0.004	0.004
DenseNet121	DL	0.0046	0	0	0	0.012	0.011
	CBR	***0.0038***	0.008	0.004	0.002	0.002	0.003

Standard Deviation on Testing Set

Model	Classifier	Avg. St. Dev.	# of Training Examples 512	1024	2048	4096	8192
VGG	DL	0.0180	0.025	0.019	0.015	0.019	0.012
	CBR	***0.0164***	0.033	0.024	0.009	0.008	0.008
Inception V3	DL	0.0126	0.019	0.010	0.013	0.013	0.008
	CBR	***0.0058***	0.011	0.007	0.005	0.003	0.003
DenseNet121	DL	0.0164	0.018	0.011	0.010	0.024	0.019
	CBR	***0.0078***	0.014	0.010	0.006	0.005	0.004

Table 1: Average standard deviation values for each DL model, organized by train and test evaluation data and then for DL and case-based classifiers. Supporting raw standard deviation values are similarly organized and arranged by number of training examples. Best (smallest) average values for each model per evaluation set are emphasized.

for Inception and DenseNet when given larger training set sizes and tested on an independent test set. However, we also see examples in which the same models show close accuracy correlation across the board with their end-to-end counterparts when evaluated on both the training set and the independent test set.

The circumstances under which such DL-CBR models may outperform end-to-end models is an interesting question. Ideally, using pretrained models creates a "quality floor" for extracted features, so it is possible that better performance is achieved because the CBR system might be able to reason more effectively from features extracted from a pretrained model that is specialized on a small training set, whereas an analogous pure DL system might require more training data/iterations to reach a similar competency. We note that it appears not to align as well with VGG feature extraction results; this may be due to the relative simplicity/shallowness of the VGG architecture.

The evaluation on the training set was conducted to give an informal "upper bound" indication of how well the methods could capture the data. Intuitively, the training accuracy for an end-to-end DL model might be artificially high due to overfitting on small training sets; this would explain why model training accuracy generally goes down as the amount of training data increases. By contrast, training accuracy values for case-based classifiers using DL features are almost all monotonically increasing with number of training examples (Figure 3). We suspect that this is another manifestation of the expected benefits of CBR for small data, though that accuracy is often highest for larger numbers of training examples suggests that a "happy medium" number of training examples may exist that produces a relative maximum training accuracy (e.g., as evidenced in the DenseNet extractor trend line, which exhibits the only decrease in training accuracy as training set size increases among case-based classifiers, Figure 3).

6 Future Research

In the reported experiments, case-based classification using DL-extracted features provided accuracy comparable or superior to the corresponding DL classifiers for image classification. A next step is to solidify these findings with tests across multiple domains and DL architectures.

We are also interested in how training of the DL feature extractor may be better aligned to the needs of the case-based classifier. This could be achieved through in some way incorporating case-based classification error into the back-propagation loss to sensitize DL feature selection to the needs of case-based classification. In addition, where knowledge-engineered features are available to the DL-CBR system [27, 40], it may be possible to provide such features to the DL model during training, potentially influencing the DL model to generate features complementary to the knowledge-engineered set. Finally, transfer learning could be applied to the Multi-Net feature extraction architecture [26], to increase accuracy through local feature selection.

7 Conclusions

This paper tests the benefit of using transfer learning when extracting features from DL networks for case retrieval. It presents an evaluation of a selection of pretrained DL models for CBR feature extraction for retrieval; the corresponding results are compared against each other to establish a best-performing architecture, as well as against earlier work on training the same models from scratch [27]. From these comparisons, we conclude that pretraining can significantly improve feature quality over training from scratch on small datasets. Among pretrained models, DenseNet appears to be the best architecture for feature extraction, and using pretraining for feature extraction for the case-based classifier often outperforms the corresponding end-to-end DL classifier (in which instances, the interpretability of case-based classification is a bonus).

We are optimistic that closer DL and CBR system integration for feature extraction, such as incorporating existing knowledge-engineered features into the network directly during training or incorporating case-based classification performance as a loss term during backpropagation, will further improve the resulting classification accuracy. We also intend to revisit our work on localized feature generation [26] with the goal of improving that model using transfer learning. It will be important to confirm the conclusions from this case study through explorations with other domain data, as well as other DL architectures.

8 Acknowledgments

This work was funded by the US Department of Defense (Contract W52P1J2093009), and by the Department of the Navy, Office of Naval Research (Award N00014-19-1-2655). Authors Wilkerson and Vats led and contributed equivalently to system formulation/experiments, Wilkerson and Leake to paper formulation, and Crandall provided project guidance. We thank Karan Acharya for early contributions.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications **7**(1), 39–52 (1994)
2. 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., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), https://www.tensorflow.org/, software available from tensorflow.org
3. Amin, K., Kapetanakis, S., Althoff, K.D., Dengel, A., Petridis, M.: Answering with cases: A CBR approach to deep learning. In: Case-Based Reasoning Research and Development, ICCBR 2018. pp. 15–27. Springer International Publishing, Cham (2018)
4. Amin, K., Lancaster, G., Kapetanakis, S., Althoff, K.D., Dengel, A., Petridis, M.: Advanced similarity measures using word embeddings and siamese networks in CBR. In: Intelligent Systems and Applications. pp. 449–462. Springer, Cham (2020)
5. Bach, K., Mork, P.: On the explanation of similarity for developing and deploying CBR systems. In: Proceedings of the Thirty-Third International Florida Artificial Intelligence Research Society Conference (FLAIRS 2020) (05 2020)
6. Barletta, R., Mark, W.: Explanation-based indexing of cases. In: Kolodner, J. (ed.) Proceedings of a Workshop on Case-Based Reasoning. pp. 50–60. DARPA, Morgan Kaufmann, Palo Alto (1988)
7. Barnett, A.J., Schwartz, F.R., Tao, C., Chen, C., Yinhao, R., Lo, J.Y., Rudin, C.: Interpretable mammographic image classification using case-based reasoning and deep learning. In: Proceedings of IJCAI-21 Workshop on Deep Learning, Case-Based Reasoning, and AutoML (2021), https://arxiv.org/pdf/2107.05605
8. Bhatta, S., Goel, A.: Model-based learning of structural indices to design cases. In: Proceedings of the IJCAI-93 Workshop on Reuse of Design. pp. A1–A13. IJCAI, Chambery, France (1993)
9. Bonzano, A., Cunningham, P., Smyth, B.: Using introspective learning to improve retrieval in CBR: A case study in air traffic control. In: Case-Based Reasoning Research and Development, ICCBR 1997. pp. 291–302. Springer, Berlin (1997)

10. Chai, J., Zeng, H., Li, A., Ngai, E.W.: Deep learning in computer vision: A critical review of emerging techniques and application scenarios. Machine Learning with Applications **6**, 100134 (2021)
11. Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., Su, J.K.: This looks like that: Deep learning for interpretable image recognition. In: Advances in Neural Information Processing Systems 32, pp. 8930–8941. Curran (2019)
12. Cox, M., Ram, A.: Introspective multistrategy learning: On the construction of learning strategies. Artificial Intelligence **112**(1-2), 1–55 (1999)
13. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009)
14. Domeshek, E.: Indexing stories as social advice. In: Proceedings of the Ninth National Conference on Artificial Intelligence. pp. 16–21. AAAI Press, Menlo Park, CA (1991)
15. Fox, S., Leake, D.: Introspective reasoning for index refinement in case-based reasoning. The Journal of Experimental and Theoretical Artificial Intelligence **13**(1), 63–88 (2001)
16. Gates, L., Leake, D., Wilkerson, K.: Cases are king: A user study of case presentation to explain cbr decisions. In: Case-Based Reasoning Research and Development, ICCBR 2023. pp. 153–168. Springer (2023)
17. Goldstein, E., Kedar, S., Bareiss, R.: Easing the creation of a multipurpose case library. In: Proceedings of the AAAI-93 Workshop on Case-Based Reasoning. pp. 12–18. AAAI Press, Menlo Park, CA (June 1993)
18. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks (2016), https://arxiv.org/abs/1608.06993
19. Kenny, E.M., Keane, M.T.: Twin-systems to explain artificial neural networks using case-based reasoning: Comparative tests of feature-weighting methods in ANN-CBR twins for XAI. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (2019)
20. Khan, A., Sohail, A., Zahoora, U., Qureshi, A.S.: A survey of the recent architectures of deep convolutional neural networks. Artificial Intelligence Review **53**, 5455 – 5516 (2019)
21. Klenk, M., Aha, D.W., Molineaux, M.: The case for case-based transfer learning. AI Magazine **32**(1), 54–69 (2011)
22. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: Proceedings of the 32nd International Conference on Machine Learning (2015)
23. Leake, D.: An indexing vocabulary for case-based explanation. In: Proceedings of the Ninth National Conference on Artificial Intelligence. pp. 10–15. AAAI Press, Menlo Park, CA (July 1991)
24. Leake, D.: CBR in context: The present and future. In: Leake, D. (ed.) Case-Based Reasoning: Experiences, Lessons, and Future Directions, pp. 3–30. AAAI Press, Menlo Park, CA (1996), http://www.cs.indiana.edu/~leake/papers/a-96-01.html
25. Leake, D., Sooriamurthi, R.: Case dispatching versus case-base merging: When MCBR matters. International Journal of Artificial Intelligence Tools **13**(1), 237–254 (2004)
26. Leake, D., Wilkerson, Z., Crandall, D.: Extracting case indices from convolutional neural networks: A comparative study. In: Case-Based Reasoning Research and Development, ICCBR 2022 (2022)

27. Leake, D., Wilkerson, Z., Vats, V., Acharya, K., Crandall, D.: Examining the impact of network architecture on extracted feature quality for CBR. In: Case-Based Reasoning Research and Development, ICCBR 2023. Springer (2023)
28. Leake, D., Ye, X.: Harmonizing case retrieval and adaptation with alternating optimization. In: Case-Based Reasoning Research and Development - ICCBR 2021. pp. 125–139. Springer, Cham (2021)
29. Li, O., Liu, H., Chen, C., Rudin, C.: Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. ArXiv **abs/1710.04806** (2018)
30. Martin, K., Wiratunga, N., Sani, S., Massie, S., Clos, J.: A convolutonal siamese network for developing similarity knowledge in the Selfback dataset. In: Proceedings of the International Conference on Case-Based Reasoning Workshops, CEUR Workshop Proceedings, ICCBR. pp. 85–94 (2017)
31. Oehlmann, R., Edwards, P., Sleeman, D.: Changing the viewpoint: Re-indexing by introspective questioning. In: Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society. Atlanta, GA (1994)
32. Ribani, R., Marengoni, M.: A survey of transfer learning for convolutional neural networks. In: 2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T). pp. 47–57 (2019). https://doi.org/10.1109/SIBGRAPI-T.2019.00010
33. Sani, S., Wiratunga, N., Massie, S.: Learning deep features for knn-based human activity recognition. In: 25th International conference on case-based reasoning (ICCBR 2017) (2017)
34. Sani, S., Wiratunga, N., Massie, S., Cooper, K.: Personalised human activity recognition using matching networks. In: Case-Based Reasoning Research and Development. Springer International Publishing (2018)
35. Schank, R., Brand, M., Burke, R., Domeshek, E., Edelson, D., Ferguson, W., Freed, M., Jona, M., Krulwich, B., Ohmayo, E., Osgood, R., Pryor, L.: Towards a general content theory of indices. In: Proceedings of the 1990 AAAI spring symposium on Case-Based Reasoning. AAAI Press, Menlo Park, CA (1990)
36. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H.S., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018)
37. Turner, J.T., Floyd, M.W., Gupta, K.M., Aha, D.W.: Novel object discovery using case-based reasoning and convolutional neural networks. In: Case-Based Reasoning Research and Development, ICCBR 2018. pp. 399–414 (2018)
38. Turner, J.T., Floyd, M.W., Gupta, K.M., Oates, T.: NOD-CC: A hybrid CBR-CNN architecture for novel object discovery. In: Case-Based Reasoning Research and Development, ICCBR 2019. pp. 373–387 (2019)
39. Wettschereck, D., Aha, D., Mohri, T.: A review and empirical evaluation of feature-weighting methods for a class of lazy learning algorithms. Artificial Intelligence Review **11**(1-5), 273–314 (1997)
40. Wilkerson, Z., Leake, D., Crandall, D.: On combining knowledge-engineered and network-extracted features for retrieval. In: Case-Based Reasoning Research and Development, ICCBR 2021. pp. 248–262 (2021)
41. Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. In: IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI) (2018)
42. Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., He, Q.: A comprehensive survey on transfer learning. Proceedings of the IEEE **109**(1), 43–76 (2021)