Extracting Case Indices from Convolutional Neural Networks: A Comparative Study

David Leake, Zachary Wilkerson(✉), and David Crandall

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

Abstract. Machine learning for extracting case features can provide great benefit over feature engineering for retrieval in poorly understood or hard to characterize domains. The effectiveness of machine learning with deep neural networks has prompted much interest in neural network approaches to feature learning in case-based reasoning, with several works showing the value of feature extraction from input data using convolutional neural networks. Those approaches are based on plausible assumptions about where in the networks to extract features for maximal usefulness. This paper presents an empirical evaluation of those underlying assumptions. We compare three extraction approaches, for an image classification task: the most common feature extraction method, extracting after the convolution layer; a recently proposed alternative, extracting after the densely-connected layers; and a new approach, extracting after the densely-connected layers using multiple networks. Our results show that the latter two approaches substantially increase case retrieval accuracy in example-sparse domains, to which case-based reasoning systems are commonly applied.

Keywords: Case-based reasoning · Deep learning · Feature learning · Hybrid systems · Indexing · Integrated systems · Retrieval

1 Introduction

Effective case-based reasoning (CBR) requires high-quality retrieval. Retrieval quality in turn generally depends on case indexing, using atomic features and possibly more complex indexing structures to characterize cases. Indexing knowledge may be acquired manually through knowledge acquisition and engineering (e.g., [7,16,20]). The manual acquisition process can provide high quality indices, but it can be expensive and is not always feasible. For example, even domain experts may not be capable of providing comprehensive feature vocabularies for poorly-understood domains or for tasks such as image recognition.

Given the effectiveness of deep learning (DL) at extracting features from data, it is natural to consider how automated indexing based on DL might supplement

M. T. Keane and N. Wiratunga (Eds.): ICCBR 2022, LNAI 13405, pp. 81–95, 2022.
https://doi.org/10.1007/978-3-031-14923-8_6

or replace human feature engineering. In CBR research, substantial attention has been focused on using convolutional neural networks (CNNs) to extract feature information from multi-dimensional raw input data. For example, CNNs have been used for extracting features from images for classifying examples with novel classes [23,24], and from outputs of three-dimensional movement sensors for human activity recognition in digital health technologies [19]. As exemplified in research by Turner et al. [23,24] and Sani et al. [19], values from feature vectors created by passing the raw input data through convolution and pooling layers early on in the network can be extracted before those vectors are further processed by densely-connected classification layers; these values become the feature set for similarity metrics in the CBR component of the hybrid system.

Ideally, the convolution and pooling steps in a CNN capture the most salient input features and structures (e.g., for image data, features such as shapes, edges, etc.). The outputs from these steps traditionally are conceptualized as the atomic features of the input. This contrasts with the outputs of the densely-connected hidden layers, which "mix and match" these features to facilitate classification. As convolution and pooling steps theoretically highlight atomic features of an input image, especially in the context of the rest of the network, it is appealing to map their features into similarity features for CBR, which has led to the use of this mapping for feature extraction.

However, it is possible that mappings at other, less-explored locations in the CNN might produce more useful features for similarity assessment in CBR. For example, the fact that features are combined in the densely-connected layers and that the final classification depends on the outputs of such combinations suggests that useful features might be extracted from the densely-connected layers in a CNN. This was proposed by Kenny and Keane [13] in the context of extracting and modifying features to generate counterfactual explanations. We hypothesize that feature extraction from the densely-connected layers will improve feature quality for the goal of increasing accuracy of case-based image classification.

This paper compares three methods that extract information in different ways from a single testbed network architecture. The extraction methods include (1) a traditional technique that extracts output values from after the convolution and pooling layers as features [19,23,24], (2) a previously proposed but less deeply-explored method that extracts the result of processing by the densely-connected layers—the inputs to the output layer of the CNN—as features [13], and (3) a novel approach that extracts as features the inputs to the output layer of n binary CNN classifiers (as opposed to one n-class classifier). Results show that extracting from before the output layer and from multiple binary classifiers can produce features that enable superior case-based classification accuracy, with especially notable performance improvements for lower-dimensional feature spaces and more class-dense scenarios.

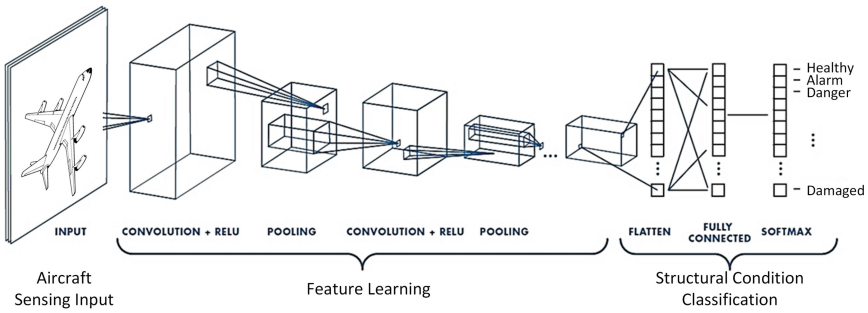

Fig. 1. Procedural diagram for a CNN in an aircraft sensor domain. Figure by Iuliana Tabian, Hailing Fu, and Zahra Sharif Khodaei is licensed under CC BY 4.0 [22].

2 Potential Feature Extraction Points in CNNs

CNNs refine the information from raw input data into a classification prediction through multiple layers. At the highest level of abstraction, a CNN may be divided into two sections. The feature extraction section consists of a series of layers designed to process the multi-dimensional raw data into a set of features passed to the classification section, which is a multilayer perceptron network (Fig. 1).

The feature extraction section typically consists of convolution and pooling layers that iteratively transform values from multi-dimensional input data. Each convolution layer consists of several different filters that are applied iteratively across the input to produce feature map output values. For each unit (e.g., pixel) in the input P indexed by (i, j), and for a filter F of size $(2k + 1) \times (2l + 1)$ (generally $k = l$), the output O is calculated by:

$$O_{ij} = \sum_{m=-k}^{k} \sum_{n=-l}^{l} F_{mn} * P_{(i-m)(j-n)} \tag{1}$$

Thus, the output of each convolution layer is a set of modified feature maps, one for each filter in the layer. In pooling steps, the resolution of each feature map generated by the preceding convolution layer is reduced by replacing each unit in a non-overlapping $r \times s$ region (again, usually $r = s$) by the value of a representative unit from the region. For the post-convolution method described below, features are extracted following the last pooling layer.

The resulting feature maps from all convolution and pooling layers in the feature extraction section are then flattened into a one-dimensional feature vector and passed as inputs into the classification section. The features are passed through multiple fully-connected dense layers until the final outputs are used as inputs to the output layer. The same final outputs are extracted as features for the post-dense and multi-net methods described below.

3 Related Work

Case retrieval quality is critically dependent on the quality of the indices used [7,14,16,17,20]. Feature vocabularies form the foundation of case indices and are commonly generated through knowledge engineering processes, reflecting comprehensive domain analysis [7,16,20]. However, manually generating the right set of features can be costly. In addition, the resulting feature set may be incomplete or unreliable when domain knowledge is imperfect. Symbolic learning methods have long been applied to refining feature selection and weighting (e.g., [1,3–5,8]), and recent work has begun to explore extraction of features and feature weights from deep neural networks. For example, Grace et al. [9] use a DL system in a recipe design domain to identify ingredient associations from the case base, and the DL system uses this information to retrieve example cases that address competing creativity and plausibility criteria. Shin et al. [21] apply an artificial neural network architecture to learn feature weightings for a CBR-based data mining task. Turner et al. [23,24] leverage CNN-generated features in a CBR classifier that classifies images for which the CNN has low confidence, defining an implicit class of images that may fall outside of known classes. Their method extracts features from different network architectures analogously to our post-convolution method, illustrating how various CNN models may be leveraged to generate CBR feature information. Sani et al. [19] apply CNNs to process multi-dimensional data from tri-axis sensors that measure human movement. Features extracted from after the convolution and pooling layers in their model are then used to classify the type and intensity of the activity using CBR retrieval.

Feature extraction from networks has also been pursued in the context of explainable AI. Kenny and Keane propose the use of CBR "twin systems", which explain network outputs by presenting cases retrieved using information extracted from the networks [12]. They also study the use of extracted feature information to generate counterfactual cases, with a method that extracts features from the output of the densely-connected layers of a CNN [13], a concept that we explore in this paper for extracting features for classification. Graziani et al. [10] use regression analyses to select entries from a field of potential concepts to identify those that a given neural network system is most likely to be learning; to achieve this mapping, the regression algorithm draws from feature data extracted from multiple regions in the network architecture.

One of the motivations for our work is the integration of knowledge-engineered and network-generated features. Weber et al. [25] leverage additional knowledge-engineered features to augment network-based explanation selection, and Barnett et al. [2] integrate interpretable, CBR-derived principles directly into a CNN image classifier. Specifically, in Barnett et al.'s work, network information is funneled toward sub-sections of the network architecture represented by prototype images in a way reminiscent of CBR retrieval. Wilkerson, Leake and Crandall [26] explore feature and weight learning using a CNN system to extract information from images to augment knowledge-engineered features. That work focuses on how learned features and knowledge-engineered features can be used in concert for greater retrieval quality and echoes the feature extraction design

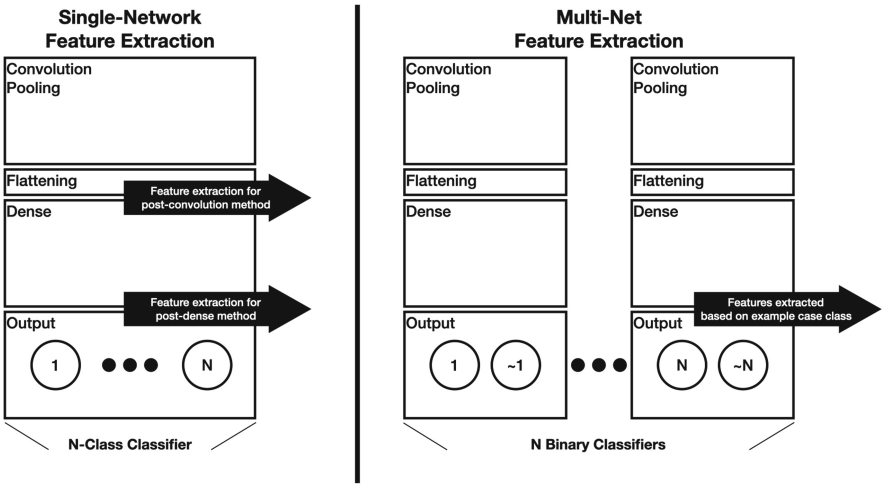

Fig. 2. Feature extraction locations for post-convolution, post-dense (both left), and multi-net (right) feature extraction methods.

of Kenny and Keane [13]. This paper solely explores feature extraction from networks.

4 Three Structure-Based Feature Extraction Methods

This paper examines three approaches for extracting CBR features/indices from a CNN architecture: one well-studied prior method [19,23,24], one method that has been explored for explanation [13] but, to our knowledge, not for classification, and a new approach proposed in this paper:

1. **Post-convolution:** Extracting feature values from between the feature extraction and classification sections in the CNN architecture
2. **Post-dense:** Extracting feature values from the outputs of the densely-connected hidden layers in the CNN's classification section
3. **Multi-net:** Expanding the single CNN architecture into multiple binary CNNs for each prediction class, and extracting features as in the post-dense approach

Figure 2 illustrates the architectures and extraction locations for each of these approaches.

4.1 Post-convolution Feature Extraction

As described in Sect. 2, iterations of convolution and pooling steps ideally remove noise and highlight atomic features from raw input data before passing the resulting output to the classification section. Therefore, if we assume that these output

values represent the atomic features of the original data (e.g., as in [19,23,24]), features extracted from between the feature extraction and classification sections (i.e., features in the flattened feature vector passed into the dense layers) are appropriate features to provide to the CBR retrieval process. The retrieval process can then use the values of those features as inputs to, for example, the distance calculation for k-nearest neighbors.

4.2 Post-dense Feature Extraction

The post-dense method extracts as features the outputs from the last densely-connected hidden layer in the classification section (i.e., the inputs to the output layer in the CNN). This design is inspired by the observation that if the feature extraction section generates atomic features from raw data, then the classification section aggregates/combines those features into richer structures more directly relevant to the classification task. From the perspective of index extraction, those structures might correspond to richer indices with more useful information. This method has not received as much attention as the post-convolution method in DL-CBR hybrid research, but some previous works have begun to explore this approach (e.g.,. [13,26]).

4.3 Multi-net Feature Extraction

During retrieval, feature applicability for classification can depend on the class to which a query case is being compared. For example, features corresponding to airplanes in an image are much more useful when classifying an airport than a library. Traditionally, the issue of relative feature importance is addressed through feature weighting, but sensitizing the feature space itself may be similarly effective. Consequently, feature extraction that enables generating local features, rather than requiring that the same features be used across the entire space, might be more useful for case indexing.

To generate local features, we propose a novel feature extraction approach, multi-net feature extraction. Rather than training a single n-class CNN classifier and extracting features, multi-net feature extraction is based on training n binary CNN classifiers that each distinguish between examples of a unique class and all examples that are not in that class. Multi-net extraction creates a unique feature generation approach for each class, and it uses different feature values based on the class of the candidate case to which the query case is compared (i.e., calculating similarity as if the query and candidate are both members of the same class). This potentially provides a benefit analagous to local similarity measures (e.g., [18]), but rather than resulting in locally adjusted feature weights, it results in an adjusted feature vocabulary. This increases flexibility by enabling richer or different representations when needed. A tradeoff of this method is increased processing cost: the training time for multi-net is increased by roughly a factor of n relative to a standard CNN architecture.

5 Evaluation

Our evaluation compares the quality of features extracted by the methods described previously, based on classification accuracy using the extracted features.

5.1 Hypotheses

Our evaluation tests the following hypotheses:

1. **Post-convolution feature extraction will lead to the weakest accuracy.** As discussed, the post-convolution method can be seen as extracting a set of initial atomic features, rather than the more refined/complex features generated by extracting from later in the network.
2. **Post-dense feature extraction will result in higher accuracy than post-convolution methods.** As discussed, the features from post-dense extraction can be seen as representing a richer range of factors.
3. **Multi-net feature extraction will yield the highest retrieval accuracy.** As discussed, the quality of the multi-net features can benefit both from the richer features of the post-dense method and from the flexibility of using different feature sets for different classes.

5.2 Test Domain and Test Set Selection

The approaches are evaluated on the Places data set for image recognition, which has been used as a standard for competition for DL-based image recognition algorithms [27]. This data set consists of images representing various common locations (e.g., alley, library, airport, etc.).

To test the approaches across problems with different numbers of classes, we generate three distinct subsets of the raw data set, respectively containing image examples from ten, twenty-five, or fifty classes. Classes for each subset are selected randomly, and for each experiment, these class subsets are frozen for consistency between iterations. In each experimental iteration, training images are selected randomly from the classes represented in the subset to create the training set. To build the training sets, an equal number N of images is chosen from each class represented in the set. Two groups of experiments are conducted. The first group keeps constant the number of examples per class regardless of the number of classes (resulting in the system having more total examples when there are more classes). The second keeps the total number of examples constant by decreasing the number of examples per class as the number of classes is increased.

For each group, three values of N are considered—10, 20, and 50. In the first group, $N = 10$ and all experiments use the same number of images per class, regardless of the number of classes. In the second group, the value of N depends on the number of classes; in the fifty-class experiment, $N = 10$, for twenty-five, $N = 20$, and for ten, $N = 50$, so that 500 training images are used for each experiment. Because CBR systems are often used in example-sparse scenarios, training set sizes are purposefully kept small in our experiments.

5.3 Testbed System

As this work only pertains to retrieval, the testbed case-based classifier has no adaptation component. The classifier performs retrieval using 1-NN and a Euclidean distance metric to determine case similarity. For these experiments, all features are weighted equally, but future work could assess the effect of feature weighting or extraction of both features and weights from CNN systems.

The CNN used to test each of the three approaches has the same structure. It derives closely from the AlexNet architecture [15] but deviates from AlexNet in excluding the bias term for output layer neurons. The rationale for this change is that the bias term influences the other input values during training but is not extracted along with other features. Consequently, a bias term could affect feature values in the CNN but remain unaccounted for when those features are transferred to the CBR system, reducing the ability of the extracted features to truly reflect the CNN's behavior.

During experimentation involving post-convolution feature extraction, the number of filters in the last convolution layer is modified to vary the number of features extracted for the CBR system; for the other two approaches, the number of neurons in the dense layers is similarly modified for the same purpose.

The activation function for all dense layers is RELU, and, as our previous work using post-dense extraction showed that training for fifty epochs produced the best results [26], we continue that training structure for this work. Each experiment iteration (i.e., training, feature extraction, and CBR-based retrieval testing) is repeated thirty times, recording mean and standard deviation values. For some multi-net tests involving the largest numbers of features, the tests terminated prematurely due to memory constraints, resulting in only twenty or twenty-five iterations overall. However, any impact would be felt only at the rightmost data point in the graph, and the results remain consistent with observed trends.

5.4 Accuracy Testing and Informal Upper Bound

In the evaluation, accuracy values are calculated by leave-one-out testing performed on the training set. These values show the relative performance of the three feature extraction methods. The figures with accuracy results also show the performance of our CNN architecture, trained on the training set—as done to generate the features extracted for the CBR system—and also tested on the *training set*. These results give an informal indication of an upper bound performance—the best performance that could result from the features available to the CBR system, were they applied in a neural network architecture to the data from which they were generated. This can be taken to roughly reflect the predictive power of the feature set under ideal conditions.

Results for the CNN upper bound should not be taken as suggesting that CNNs necessarily outperform CBR in this task, for two reasons. First, the CNN was trained on all examples, including each query being processed; in contrast, the CBR systems process each query with its corresponding case omitted from

the case base. Second, the strongest CBR performance would require tuning similarity weighting, which is not done in our experiments.

6 Results and Discussion

6.1 Comparative Performance

Figures 3 and 4 show training accuracy versus number of features for different numbers of classes, comparing the three methods and the CNN classifier. In general, the post-dense approach significantly outperforms the post-convolution extraction method, and in many instances, the multi-net method outperforms both the post-dense and post-convolution methods, especially for smaller numbers of features. There are only a few instances in which the post-convolution method rivals either novel method, and only for limited numbers of features. The overall pattern supports the three hypotheses and suggests that the novel approaches improve feature quality. The results also illustrate several tradeoffs:

When the Total Number of Examples Varies, There Is a Tradeoff Between More Classes Increasing the Number of Training Examples and Increasing the Degrees of Freedom: Especially in the trends for post-dense extraction, the maximum accuracy values are highest in the 25-class case when holding the number of training examples per class constant. This illustrates a tradeoff between the number of classes and the number of examples per class. Specifically, a larger number of classes affords a greater number of training examples overall but creates more degrees of freedom in the classification problem itself. The opposite is true when the number of classes is reduced. Thus, we see a local maximum in the 25-class data for the post-dense results, as it represents a "happy medium" between these two factors. This is further supported by results when the number of training examples is held constant (Fig. 4). In this instance, accuracy curves for each approach essentially parallel one another across the different numbers of classes, with overall decreases in accuracy with a higher number of classes easily attributable to fewer training examples per class.

Fewer Features Can Harm CNN convergence, while many features can lead to a "curse of dimensionality": For each method (except the CNN classifier), accuracy broadly decreases as the number of features increases. Also evident in the post-dense method's accuracy curves, an even more pronounced decrease in accuracy occurs for small numbers of features (we hypothesize that a similar trend exists for the multi-net method for smaller numbers of features than we show here). We believe that the first phenomenon is a consequence of the "curse of dimensionality", with individual features having increasingly small influence on distances between examples in feature-dense spaces. Note that we do not see this in the CNN classifier, as CNNs tend to be robust to (and often more performant with) large numbers of features. Relative to the second phenomenon, if a neural network has access to too few parameters during training,

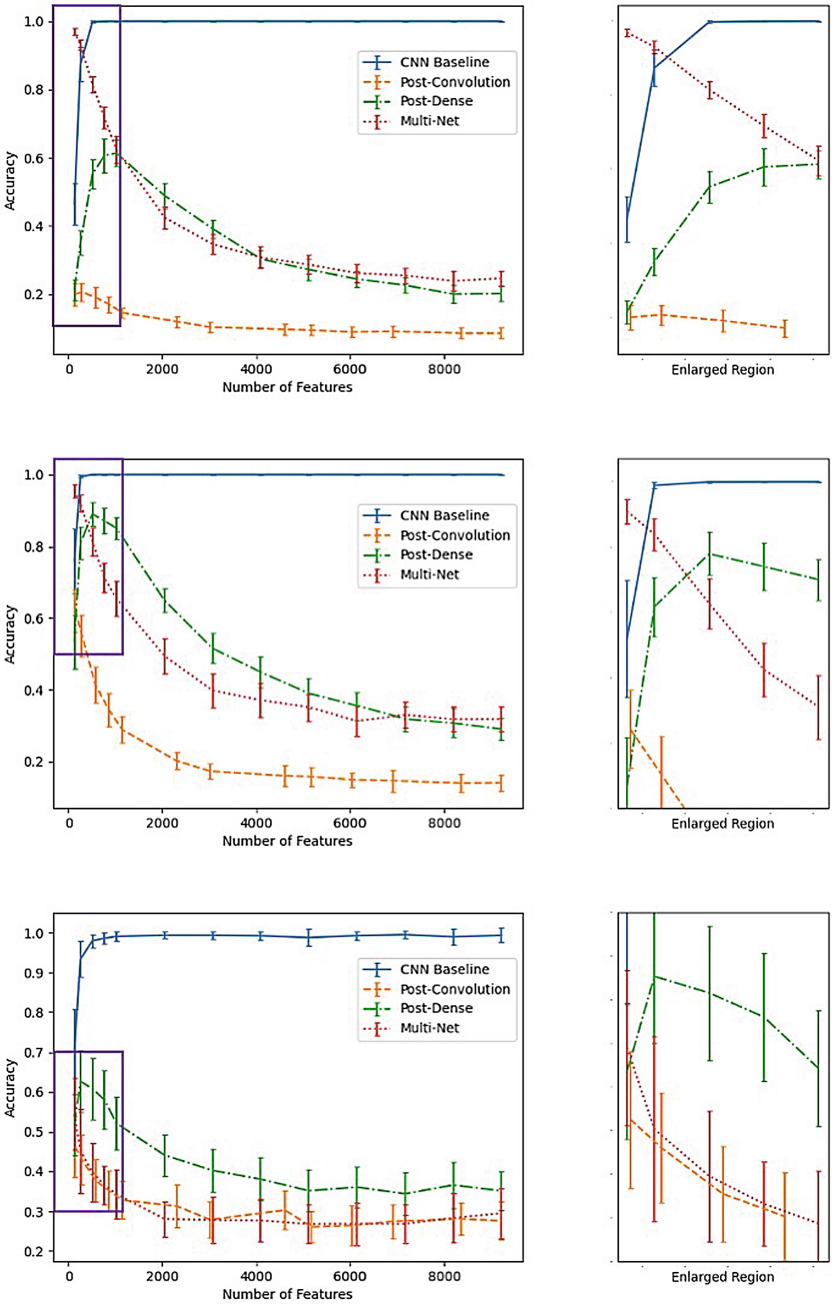

Fig. 3. Accuracy versus number of features for the 50-class case (top), 25-class case (middle) and 10-class case (bottom), using ten training examples per class. Error bars represent one standard deviation. Boxed regions at left on each graph are shown enlarged at right.

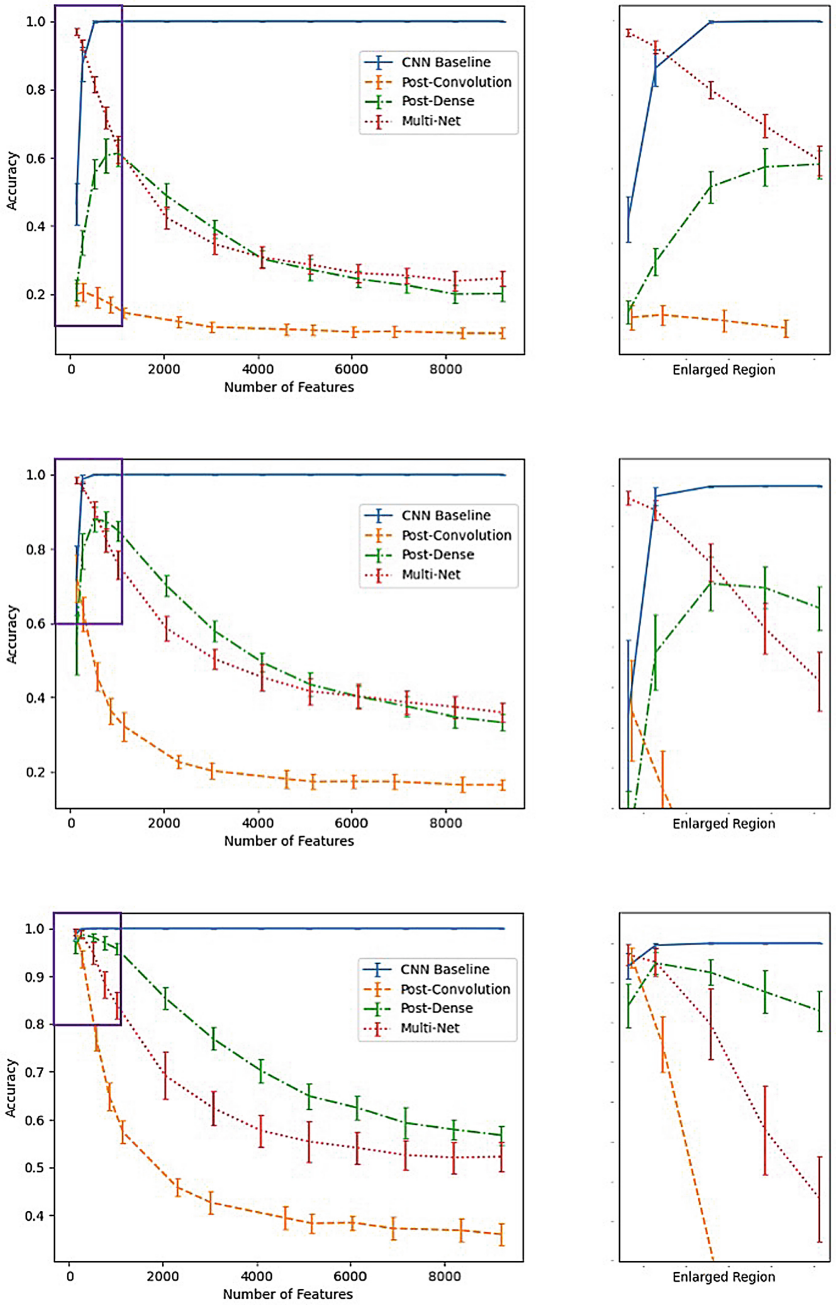

Fig. 4. Accuracy versus number of features for the 50-class case (top), 25-class case (middle) and 10-class case (bottom), using 500 training examples for each. Error bars represent one standard deviation. Boxed regions at left on each graph are shown enlarged at right.

it may not converge on a representative feature set. Thus, we hypothesize the existence of a tradeoff for the number of features in the CBR system—with too few features, retrieval performance suffers because there are not enough features for the network to converge while training, and with too many features, the distance calculations are in so many dimensions that the individual features are ineffective. Furthermore, the "happy medium" for this tradeoff should also be dependent on the number of classes (i.e., more features are required to distinguish between a larger number of classes); this assertion is supported by the data (e.g., the location of the maxima for the post-dense method).

Multi-net Has an Accuracy-Training Time Tradeoff Instead of an Accuracy-Explainability Tradeoff: The end-to-end CNN classifier, expected to be an upper bound, outperforms all methods almost all of the time in each experiment. However, surprisingly, the multi-net approach outperforms the CNN for small numbers of features. In this way, multi-net performance contrasts with the traditional conceptualization of the accuracy-explainability tradeoff [11]: compared to the CNN it trades off increased training time against accuracy, while (through the use of CBR) retaining explainability.

6.2 Discussion

A primary observation from the experiments is the superior performance of the multi-net approach. Case-based classification using the local features generated by the multi-net approach frequently outperforms the other case-based approaches, and it outperforms the CNN classifier for small numbers of features. This is reasonable because fewer degrees of freedom in doing binary rather than multiclass classification would suggest that fewer features are required to discriminate between the classes. Instead of trading off accuracy and explainability, as in other models that use extracted features for CBR, for small numbers of features multi-net trades off accuracy and training efficiency.

In addition, we hypothesize that the local feature extraction of the multi-net approach is critical to its performance. Specifically, the multi-net feature extraction model, which selects a set of features based on the class of a candidate case, can be seen as predicating features on the CBR component of the model, as a form of the traditional CBR situation assessment process, which elaborates and rerepresents features of an input query to be commensurate with the feature vocabulary of the case base. Here, instead of asking the feature extraction model "what features are present in the query?" it asks multi-net "what features related to class x are present in the query?"

It is natural to ask whether a process similar to the multi-net process could be achieved in an end-to-end CNN approach. It would be possible to use a collection of CNN binary classifiers and select the one that predicts that the query is in its class. However, in practice, multiple models may answer in the affirmative, requiring some tiebreaker (e.g., a softmax), raising the question of whether the result would be significantly different from a multiclass CNN. More research is needed to quantify the exact factors that enable multi-net's strong performance.

7 Ramifications for Interpretability

Hybrid systems as explored in this work sit at the intersection between DL and CBR systems. The nearest-neighbor retrieval and classification process presented here is interpretable in being able to present cases to explain decisions, a capability missing from DL systems alone. However, using learned features for similarity assessment makes such systems less interpretable than a CBR system applying knowledge-engineered features exclusively, for which case similarity—the reason the case was selected—could be more readily explained. However, in domains where a human user can holistically ascribe similarities between a query case and the retrieved case, this "middle ground" between DL and knowledge-engineered CBR may provide sufficient interpretability, and there is evidence for the value of case presentation alone as an explanation mechanism [6].

Furthermore, these approaches (especially multi-net) may be applicable in domains for which classification accuracy is more important than explainability, and for which minimal training data exists. However, further research is needed into the accuracy-interpretability tradeoff when using network-learned features, as well as into the potential use of methods such as counterfactual/semi-factual explanation generation [13] to generate additional cases to holistically illuminate the boundaries for cases considered similar, even if it is not possible to point to variations of user-understandable features.

8 Conclusions and Future Work

Feature extraction from neural networks can facilitate the application of case-based reasoning to hard to characterize domains, where knowledge-engineered feature information may be difficult and/or expensive to produce. Prior work in feature extraction has focused primarily on extracting features immediately after the convolution and pooling layers, based on the intuition that such layers will be best suited to feature extraction because they provide atomic descriptions of domain features. This paper challenges that assumption, proposing feature extraction after the dense layers. It presents a new multi-net approach for this extraction, and, to our knowledge, presents the first comparative evaluation to assess the performance of alternative extraction locations. The results highlight the strength of post-dense feature extraction in example-sparse domains, where CBR systems are often the method of choice; furthermore, they support that multi-net can provide even stronger performance in lower-dimensional spaces.

In future research, we expect to investigate the use of multi-net feature extraction with different CNN architectures. In addition, building on our prior work on combining network-learned and expert-provided features [26], we plan to test its effects on hybrid retrieval using both extracted features and knowledge-engineered features. We also intend to consider more example-dense training scenarios and larger datasets, as well as exploring how similarity weights might be extracted along with features from network architectures to further improve performance.

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). We thank the Indiana University Deep CBR group, especially Vibhas Vats and Lawrence Gates, for valuable discussions of this work.

References

1. 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)
2. Barnett, A.J., et al.: Interpretable mammographic image classification using case-based reasoning and deep learning. In: IJCAI Workshops 2021 (2021)
3. 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)
4. Bonzano, A., Cunningham, P., Smyth, B.: Using introspective learning to improve retrieval in CBR: a case study in air traffic control. In: Leake, D.B., Plaza, E. (eds.) ICCBR 1997. LNCS, vol. 1266, pp. 291–302. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-63233-6_500
5. Cox, M., Ram, A.: Introspective multistrategy learning: on the construction of learning strategies. Artif. Intell. **112**(1–2), 1–55 (1999)
6. Cunningham, P., Doyle, D., Loughrey, J.: An evaluation of the usefulness of case-based explanation. In: Ashley, K.D., Bridge, D.G. (eds.) ICCBR 2003. LNCS (LNAI), vol. 2689, pp. 122–130. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45006-8_12
7. 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)
8. Fox, S., Leake, D.: Introspective reasoning for index refinement in case-based reasoning. J. Exp. Theor. Artif. Intell. **13**(1), 63–88 (2001)
9. Grace, K., Maher, M.L., Wilson, D.C., Najjar, N.A.: Combining CBR and deep learning to generate surprising recipe designs. In: Goel, A., Díaz-Agudo, M.B., Roth-Berghofer, T. (eds.) ICCBR 2016. LNCS (LNAI), vol. 9969, pp. 154–169. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47096-2_11
10. Graziani, M., Andrearczyk, V., Marchand-Maillet, S., Müller, H.: Concept attribution: explaining CNN decisions to physicians. Comput. Biol. Med. **123**, 103865 (2020)
11. Gunning, D., Aha, D.W.: DARPA's explainable artificial intelligence (XAI) program. AI Mag. **40**(2), 44–58 (2019)
12. 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)
13. Kenny, E.M., Keane, M.T.: On generating plausible counterfactual and semi-factual explanations for deep learning. In: Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21), pp. 11575–11585. AAAI (2021)
14. Richter, M.M., Weber, R.O.: Relations and comparisons with other techniques. In: Case-Based Reasoning, pp. 523–538. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40167-1_23

15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems, vol. 1, pp. 1097–1105 (2012)
16. 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 (1991)
17. de Mántaras, L.R., et al.: Retrieval reuse revision and retention in CBR. Knowl. Eng. Rev. **20**(3), 215–240 (2005)
18. Ricci, F., Avesani, P.: Learning a local similarity metric for case-based reasoning. In: Veloso, M., Aamodt, A. (eds.) ICCBR 1995. LNCS, vol. 1010, pp. 301–312. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-60598-3_27
19. Sani, S., Wiratunga, N., Massie, S.: Learning deep features for kNN-based human activity recognition. In: Proceedings of ICCBR 2017 Workshops (CAW, CBRDL, PO-CBR), Doctoral Consortium, and Competitions co-located with the 25th International Conference on Case-Based Reasoning (ICCBR 2017), Trondheim, Norway, June 26–28, 2017. CEUR Workshop Proceedings, vol. 2028, pp. 95–103. CEUR-WS.org (2017)
20. Schank, R., et al.: 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)
21. Shin, C., Yun, U.T., Kim, H.K., Park, S.: A hybrid approach of neural network and memory-based learning to data mining. IEEE Trans. Neural Netw. Learn. Syst. **11**(3), 637–646 (2000)
22. Tabian, I., Fu, H., Khodaei, Z.S.: A convolutional neural network for impact detection and characterization of complex composite structures. Sensors **19**(22), 4933 (2019)
23. Turner, J.T., Floyd, M.W., Gupta, K.M., Aha, D.W.: Novel object discovery using case-based reasoning and convolutional neural networks. In: Cox, M.T., Funk, P., Begum, S. (eds.) ICCBR 2018. LNCS (LNAI), vol. 11156, pp. 399–414. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01081-2_27
24. Turner, J.T., Floyd, M.W., Gupta, K., Oates, T.: NOD-CC: a hybrid CBR-CNN architecture for novel object discovery. In: Bach, K., Marling, C. (eds.) ICCBR 2019. LNCS (LNAI), vol. 11680, pp. 373–387. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29249-2_25
25. Weber, R.O., Shrestha, M., Johs, A.J.: Knowledge-based XAI through CBR: there is more to explanations than models can tell. In: ICCBR Workshops 2021, pp. 75–86 (2021)
26. Wilkersoon, 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)
27. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. Adv. Neural Inf. Process. Syst. **27** (NIPS) (2014)