Perception Over Time: Temporal Dynamics for Robust Image Understanding Supplementary Material

Maryam Daniali Edward Kim
Drexel University
Philadelphia, PA
{maryam.daniali,edward.kim826}@drexel.edu

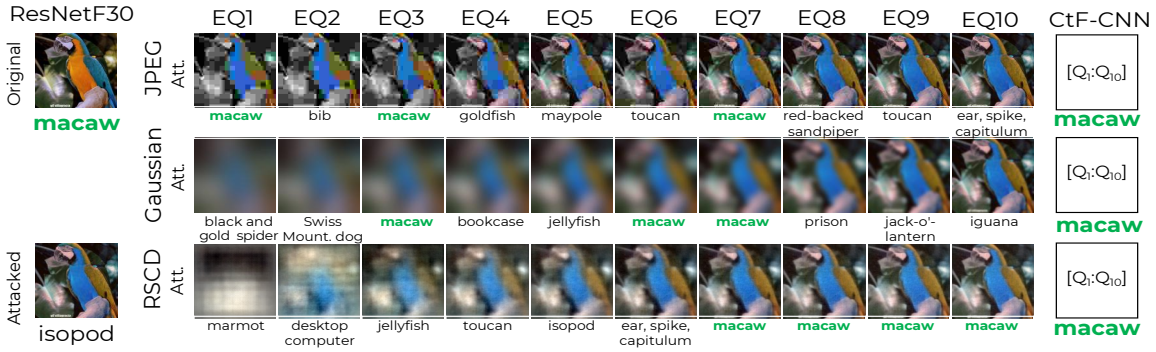


Figure 1. Classification results on a sample image from the class 'macaw' from ImageNet. Accurate detections are marked in green. On the left, we see ResNetF30 could correctly recognize the original image; however, it miss-classified the attacked version as an 'isopod.' On the right, we see the performance of Expert Models (EQ_i), ResNet fine-tuned on images of quality i, and our proposed CtF framework on the decomposed attacked images. Expert Models struggle with continuing their recognition and could miss the classification even with the correct detection of lower qualities. See EQ1, EQ3, and EQ7 using JPEG compression (first row), and EQ3, EQ6, and EQ7 using Gaussian smoothing (second row). Our proposed decomposition method, RSCD, could help Expert Models make a more stable recognition; see EQ7, EQ8, EQ9, and EQ10 using RSCD (third row). Finally, the CtF-CNN model could leverage the information generated in a course-to-fine manner and over time to better detect the object and correctly classify the attacked version using any of the decomposition methods.

1. Datasets

1.1. ImageNet10

As mentioned in the main paper, we selected 10 visually distinctive classes from ImageNet to make it practical and feasible for the human-study task. This dataset is referred to as ImageNet10 and was used in the experiments provided in Figure 2 in the main paper. The select classes are listed in Table 1.

1.2. ImageNet30

We added 20 randomly selected non-overlapping classes from ImageNet to ImageNet10 to create ImageNet30. Table 2 presents the 30 classes included in ImageNet30 with their text descriptions. The classes shared with ImageNet10 are marked in bold.

2. Human Study

For the human-study task, we used ImageNet10 and showed our participants a series of decomposed images, starting from a very blurry image, Q_1, all the way to a very detailed image, Q_{10}. At each step, the participants were tasked to type the main object they recognized in the presented quality and choose their confidence level, scoring from 1 to 5, where 1 represented "unsure," and 5 represented "certain". Also, to cover all classes of ImageNet10 for each decomposition method, we randomly chose one image per class and showed its 10 decomposed images in order. Thus, each participant answered 30 sets of unique images, each set containing 10 decomposed images, from Q_1 to Q_{10}. See Figure 2 for screenshots of the human task web application.

Since we asked our participants to type in the main ob-

Table 1. Select classes with their text descriptions in ImageNet10.

Class ID	Discription	Equivalent Options
n01443537	goldfish, Carassius auratus	fish, goldfish, fishes
n02107574	Greater Swiss Mountain dog	dog
n03180011	desktop computer	desktop, laptop, computer, desktop computer, keyboard, monitor, display, TV
n01514859	hen	hen, chicken, rooster
n01818515	macaw	parrot, bird
n02690373	airliner	airliner, airplane, plane
n02870880	bookcase	bookcase, library, book shelf, shelf, book
n03670208	limousine, limo	car, limo, limousine, vehicle
n03590841	Jack-o-lantern	pumpkin, Halloween, lantern
n13133613	ear, spike, capitulum	corn, plant, cereal, popcorn, wheat

Table 2. Select classes with their text descriptions in ImageNet30. The overlapping classes with ImageNet10 are marked in bold.

Class ID	Description	Class ID	Description
n01443537	goldfish, Carassius auratus	**n02690373**	airliner
n01496331	electric ray, crampfish, numbfish, torpedo	n02834397	bib
n01514859	hen	**n02870880**	bookcase
n01558993	robin, American robin, Turdus migratorius	**n03180011**	desktop computer
n01677366	common iguana, iguana, Iguana iguana	n03355925	flagpole, flagstaff
n01773157	black and gold garden spider, Argiope aurantia	**n03590841**	jack-o'-lantern
n01818515	macaw	**n03670208**	limousine, limo
n01843383	toucan	n03733131	maypole
n01910747	jellyfish	n03796401	moving van
n01990800	isopod	n04005630	prison, prison house
n02011460	bittern	n04263257	soup bowl
n02027492	red-backed sandpiper, dunlin, Erolia alpina	n04486054	triumphal arch
n02091635	otterhound, otter hound	n09246464	cliff, drop, drop-off
n02107574	Greater Swiss Mountain dog	n13037406	gyromitra
n02361337	marmot	**n13133613**	ear, spike, capitulum

ject they recognized at each level without any knowledge of the database — also known as an open-ended task — their responses were not limited to the 1000 available classes in ImageNet. Thus, the recognition task for human participants could be more challenging than the off-the-shelf models with only 1000 choices.

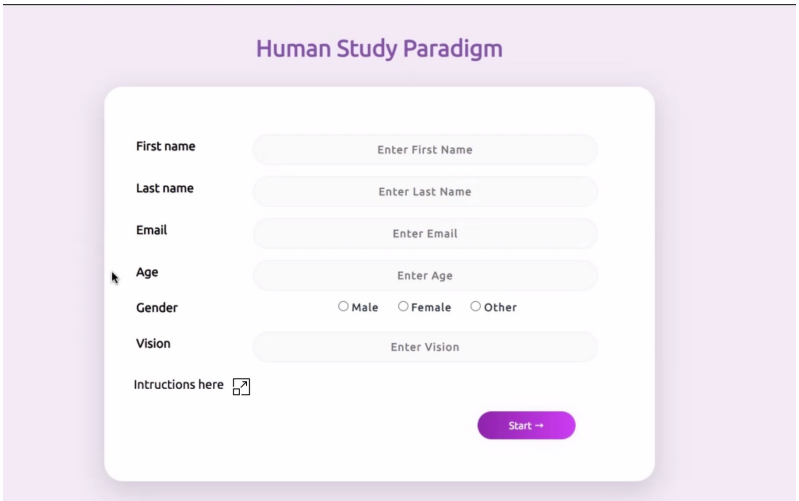
Furthermore, humans have a hierarchical model of concept classification, in the sense that they can be very general or very specific in classifying objects depending on their categories [8]. More specifically, if we take the recognition hierarchy as a pyramid, the more specific classifications, i.e., subordinate concepts, are at the top, and the more general classifications, i.e., superordinate concepts, are at the base. The concepts that are more specific than superordinate and more general than subordinate are called basic concepts.

For example, a classification like animal or mammal would be a superordinate concept. Then another level up could be classified as a dog with its specific animal features, which is a basic concept. Lastly, the subordinate category could be a dog breed.

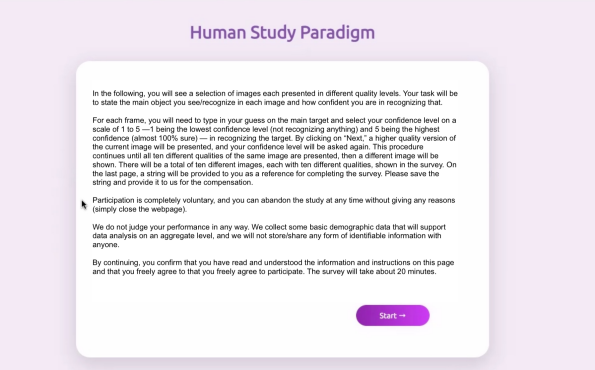
2.1. Humans' performance limitation.

Studies show humans classify objects at different levels of this hierarchy based on the order of availability and, therefore, the order of acquisition of various categorization schemes, starting with the basic level [2]. Thus, we post-processed the participants' answers to the basic level category employing the same structure used in defining class labels, and hierarchies in creating ImageNet using WordNet lexical database [5]. The corresponding categories are listed in Table 1, in Column Equivalent Options. We then used the same categories for identifying the performance of the models and compared their performance with that of the human participants. See Figures 3(c) and 3(f) in the main paper.

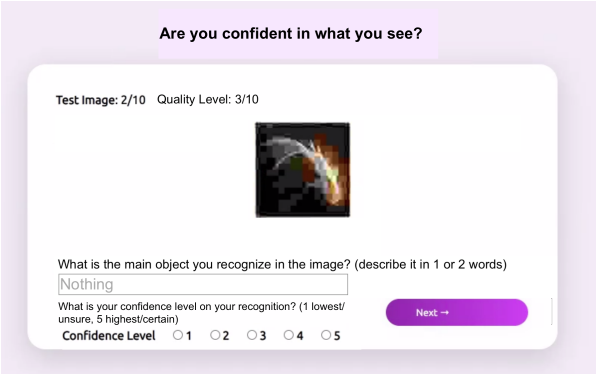
To the best of our knowledge, there is no study on evaluating human performance at the basic level on ImageNet. However, there are studies evaluating human accuracy by relying on expert annotators [10, 11]. Even with year-long



(a) First page of the human task.



(b) Instructions for the human task.



(c) Sample task image.

Figure 2. Screenshots of the human experiment procedure. (a) the first page of the web application, (b) the instructions, and (c) a sample image for the recognition task where the participants needed to type the main object they see in the provided image and their confidence in their recognition.

trained human labelers, the top1 accuracy on ImageNet is around 93% [10].

A couple of identified issues on the original database that results in a lower accuracy than 100% are as follows:

- About 20% of images have more than one valid label.
- Humans struggle with fine-grained distinctions within the 410 animal classes and 118 dog classes.
- There are some misclassified images in the ImageNet dataset. For instance, at least 4% of the birds are misclassified.
- Untrained labelers will likely be less accurate, suggesting that human robustness is a more stable property than direct accuracy measurements.

In addition to the mentioned issues on the original dataset, there are some concerns specifically related to our tasks.

- Our participants were not trained on the dataset.
- We had an open recognition task, and the participants had no guidance on the list of potential objects.
- Many images had more than one object. Thus, the participants did not know what the target object was.
- Even on the higher quality levels, some images were blurry and challenging to categorize, specifically using Gaussian smoothing and RSCD.

These concerns could justify why our untrained participants could not pass 80% accuracy. However, their performance followed our original hypothesis on the accuracy and confidence trends over time. Our results show that the confidence and accuracy of our participants increased as the quality of the images increased, and unlike the deep learning models, there was no sudden change in their accuracy or confidence; instead, their accuracy and confidence increased gradually over time and quality. See Figures 3(c) and 3(f) in the main paper.

3. Models

3.1. Model Architecture.

CtF-CNN and CtF-LSTM have a similar feature extraction part. This part takes the outputs of Expert Models, all or a selection of them, as a sequence and feeds them into two 1-D CNN layers with 128 and 64 filters, respectively. The kernel size of the first 1-D CNN layer is equal to the size of the feature vector of each Expert Model, in our case, is 1024, and the kernel size of the second 1-D CNN is half the size, in our case, 512.

For CtF-CNN, the feature extraction part is followed by three layers. A 1-D CNN with 32 filters and a kernel size of 256, followed by a global average pooling and a fully connected layer of size $N_{classes}$, where $N_{classes}$ is the number of classes in the dataset, i.e., 30 in ImageNet30.

For CtF-LSTM, the feature extraction part is followed by two layers. An LSTM layer with $N_{qualities} \times N_{classes}$ hidden cells—where $N_{qualities}$ is the number of selected Expert Models—and a fully connected layer of size $N_{classes}$. See Section 3 in the supplementary materials.

The detailed architecture of our CtF models is provided in Figure 3. We use the feature vector output of $ExpertQ_i$, where $i \in [1, 10]$, as an input to CtF-CNN and CtF-LSTM. This approach allows us to process information in a CtF manner starting from the "gist" of an image, i.e., Q_1, and all the way to its high-frequency information available in Q_{10}.

For the "gaining" and "losing" experiments, we examined various cases where the CtF input sequence was a subset of $[Q_1, Q_{10}]$, and showed the important role of each quality level in the overall performance of the CtF models. Although applying the CtF framework on many smaller subsets could outperform ResNetF30 on the original images, the best accuracy was received when all the information from Q_1 to Q_{10} was used in the CtF framework. See Figure 5 of the main paper.

3.2. Training.

During ResNetF30 fine-tuning, we used a Root Mean Squared Propagation optimizer with $\rho = 0.9$, $\mu = 0.9$, $\epsilon =$ 1e-07, and an initial learning rate of 0.001. We used L1 regularization with a regularization factor of 0.01 and trained the model for 35 epochs with a batch size of 10. For $ExpertQ_i$, we fine-tuned ResNetF30 on decomposed images of quality i from the select decomposition method. See comparisons in Section 3.5 in the main paper. We also used similar parameters used in training ResNetF30. In training CtF-CNN and CtF-LSTM, we applied batch normalization after each 1-D convolutional layer. Also, we employed an Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and an initial learning rate of 0.001. We trained both models for 35 epochs with a batch size of 10. Also, we used 75% of the data for training and the rest for testing in all models.

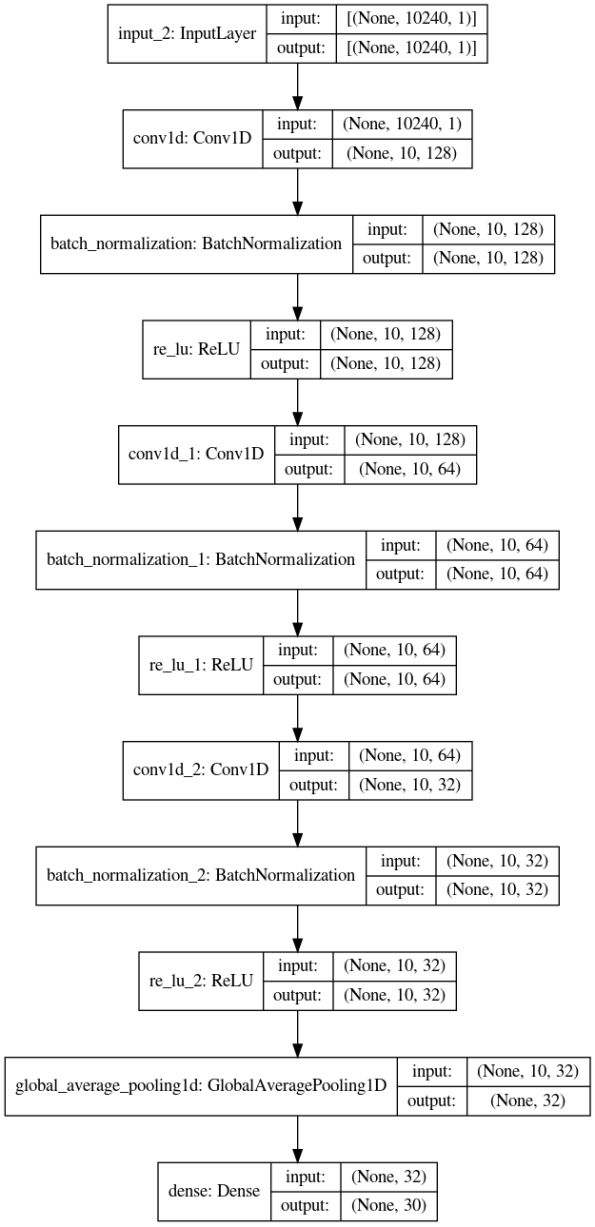
3.3. Training Parameters and Cost.

Is the CtF framework costly? We used ResNet50 pretrained weights on ImageNet for the Expert models (EQ_i) and only fine-tuned their top 3 layers (see Figure 2 in the main paper). Thus, each Expert has only 2.1 million trainable parameters. For the CtF framework, we freeze the Expert models and only train the recurrent layers following the Experts' output in the CtF model. The CtF framework has 4.7 M parameters which, altogether with the 10 Experts, is still significantly (one order of magnitude) less costly than other models that incorporate some level of feedback in their learning and classification, such as attention models with typically more than 100 million parameters.

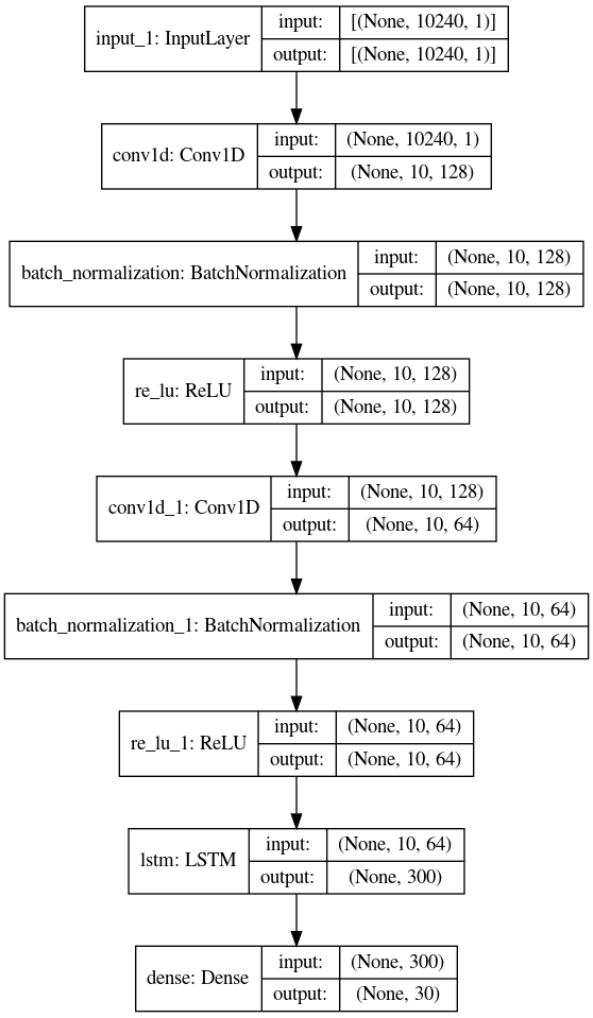
Is the RSCD decomposition costly? Similar to other recurrent generative models, such as diffusion models, sparse coding is not fast, but in contrast to them, while using its special hardware, can perform relatively fast. In fact, there are minimal computational downsides to performing our sparse decomposing approach on images. More precisely, with neuromorphic hardware, *even orders of magnitude computational- and energy-efficiency can be achieved over standard Von Neumann implementations*. First, the CtF images are not handled sequentially but rather in a single pass through the model (similar to batching [6]), resulting in only a minimal computational percentage increase proportional to the input size. Furthermore, RSCD can be very efficiently computed in both time and energy costs on spiking neural network hardware. Davies et al. have shown significant improvements in sparse coding both in energy consumption and speed [4]. As an example, Loihi is a neuromorphic many-core processor with on-chip learning. It implements a spiking neural network in silicon and demonstrates LCA on their architecture with an improvement by *48.7× energy efficiency, 118× speed, and 5760× EDP* over a CPU implemented FISTA [1].

Are JPEG and Gaussian CtF approximators more cost-efficient? JPEG and Gaussian may be preferable for Von Neuman hardware, thus may be easier to be plugged into many applications with such hardware. Also, in many of our experiments, JPEG has shown performance on par with RSCD. However, JPEG and Gaussian decomposition methods are not bio-inspired, and research has shown flaws with them [7]. We also demonstrate unsteady behaviors in JPEG and Gaussian (See V-shapes in Figure 5(b), Overconfidence in Figures 4(b) and 4(c) in the main paper, and Figure 5) while RSCD is robust.

Also, such approximators' superiority is not predictable when dealing with different attacks. For instance, we see



(a) CtF-CNN Model Architecture.



(b) CtF-LSTM Model Architecture.

Figure 3. The architecture of our CtF framework. The input layer for each model is a sequence containing the feature vector outputs of $ExpertQ_i$, where $i \in [1:10]$. Based on our specification, each feature vector is of length 1024, resulting in an input sequence of length 10240.

Gaussian performing better in Square attack while it performs worst in PGD attack (See Table 1 in the main paper). Our results demonstrate similar patterns on non-CtF models as well (See Table 2 in the main paper). Thus, in our opinion, it is important to introduce all these methods to the community and let them choose based on their application.

3.4. Experimental Environment.

We generated the RSCD images on a machine equipped with an Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz clock frequency and an NVIDIA GeForce RTX 2080 GPU. For other decomposition methods and the experiments, we used a machine equipped with an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz clock frequency and an NVIDIA GeForce GTX 1070 GPU.

The execution time for the experiments is as follows:

- Generating the RSCD images: 10 hours, 24 minutes, and 39 seconds.
- Fine-tuning ResNetF30 based on the Keras implementation of ResNet50 [3]: on average 2.7 minutes.
- Fine-tuning Expert Q_i based on the Keras implementation of ResNet50 [3]: on average 2.9 minutes.
- Training CtF-CNN using saved Expert Q_is: on average 40 seconds.
- Traning CtF-LSTM using saved Expert Q_is: on average 42 seconds.

4. Ablation Experiments

Here, we demonstrate the role of each quality level while interacting with other qualities in visual perception. The standard performance of CtF-CNN and CtF-LSTM improves while "gaining" data, see Table 3, which emphasizes the importance of fine-level input.

5. Adversarial Robustness

In this section, we present additional plots on the performance of the baselines (ResNetF30 and Expert Models), as well as the CtF models.

Figure 1 shows the classification results of the baseline models, ResNetF30, and Expert models, as well as the CtF framework on a sample image from the actual class 'macaw.' We see how baseline models, ResNetF30 and Expert Models, struggle with detecting the object on the attacked version. However, the CtF-framework, in this case, CtF-CNN, could leverage the information over time and correctly detect the class of the sample image.

As discussed in the main paper, while ResNetF30 could perform relatively well in classifying original images, Acc. = 0.78, see Table 1 in the main paper, its performance dropped significantly on the attacked image, Acc. Att. = 0.15.

In Table 4, we present the performance of ResNetF30 on the decomposed attacked images using all three decomposition methods. We see an accuracy drop even on the highest quality decompositions, Acc. Att. = 0.13 on JPEG compression with $Q10$, which shows that a conventional CNN model such as ResNetF30 is prone to some high-frequency information in addition to the attacks applied by a strong attacker such as PGD.

A model is counted as overconfident when it has high prediction confidence, the value of softmax confidence [9], but its accuracy is very low. We can also interpret the confidence of the true class as an indication of the model's correctness. Thus, a comparison between the prediction confidence and the true class can directly give us insights into the model's correctness and confidence.

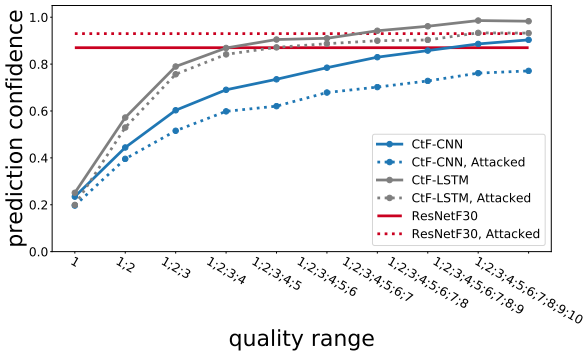
We provide comparisons on the prediction confidence and the true-class confidence of all three decomposition methods on "gaining" and "losing" data in Figures 4 and 5, respectively. These figures show a big gap between ResNetF30's prediction and true-class confidence, especially on the attacked images, which shows that the ResNetF30 model is confidently incorrect. We still see the effects of the attacks even when using the CtF framework; however, the gap between the prediction confidence and the true-class confidence is relatively minor when using CtF-CNN and CtF-LSTM, resulting in a more robust visual perception framework. We also see the effects of the artifacts in using the decomposition approximators (JPEG compression and Gaussian smoothing) in the true class confidence while losing data. See the "V" shapes in Figures 5(d) and 5(e). These results once more suggest the importance of using bio-inspired approaches in creating the CtF decompositions.

Table 3. CtF models surpass ResNetF30 while "gaining" data. Cells with the minimum quality range that top ResNetF30 (0.78) are marked in with a horizontal line. We see an increasing pattern in accuracy as the CtF models receive more information over time which is in line with our initial hypothesis as well as human results. Notably, the performance of CtF-LSTM drops suddenly when receiving information of Quality 10 based on JPEG reconstructions, indicating the instability of non-bioinspired methods such as JPEG reconstruction to generate semi-CtF decompositions.

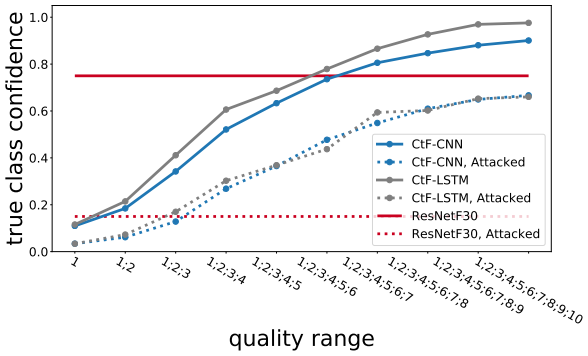
	CtF-CNN, Acc.			**CtF-LSTM, Acc.**		
Quality Range	Gaussian	JPEG	RSCD	Gaussian	JPEG	RSCD
[1]	0.200	0.373	0.160	0.184	0.363	0.147
[1; 2]	0.323	0.515	0.221	0.299	0.541	0.224
[1; 2; 3]	0.392	0.648	0.405	0.403	0.651	0.427
[1; 2; 3; 4]	0.493	0.763	0.616	0.464	0.717	0.627
[1; 2; 3; 4; 5]	0.555	**0.840**	0.739	0.539	**0.827**	0.693
[1; 2; 3; 4; 5; 6]	0.557	**0.909**	**0.872**	0.544	**0.811**	**0.800**
[1; 2; 3; 4; 5; 6; 7]	0.691	**0.923**	**0.915**	0.597	**0.917**	**0.869**
[1; 2; 3; 4; 5; 6; 7; 8]	**0.781**	**0.968**	**0.957**	0.736	**0.952**	**0.941**
[1; 2; 3; 4; 5; 6; 7; 8; 9]	**0.808**	**0.971**	**0.981**	0.728	**0.973**	**0.976**
[1; 2; 3; 4; 5; 6; 7; 8; 9; 10]	**0.923**	**0.989**	**0.989**	**0.861**	**0.925**	**0.979**

Table 4. Decomposition methods alone do not remove PGD attacks for the images to any level that immunizes ResNetF30. We do not see any improvement in the accuracy of ResNetF30 while classifying decomposed attacked images using any of the decomposition methods, even on the highest quality decompositions. Compare **max(Acc. Att. PGD Q10) = 0.131** with **Acc. Att. PGD = 0.15** from Table 1 in the main paper. These results demonstrate that these decomposition methods by themselves are not immunizing techniques. The key to immunity to such attacks in our proposed CtF framework is the convergence of perception over time and not the decomposed images themselves.

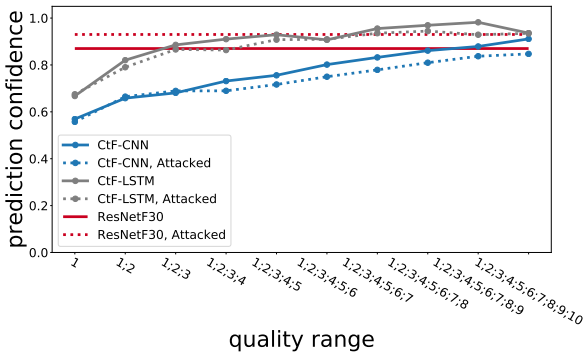
	Gaussian		**JPEG**		**RSCD**	
Input	**Acc. Att.**	**Pred. Conf.**	**Acc. Att.**	**Pred. Conf.**	**Acc. Att.**	**Pred. Conf.**
Q1	0.016	0.225	0.064	0.473	0.029	0.429
Q2	0.021	0.240	0.064	0.475	0.061	0.321
Q3	0.016	0.245	0.067	0.546	0.064	0.332
Q4	0.037	0.249	0.072	0.629	0.064	0.421
Q5	0.048	0.255	0.085	0.663	0.075	0.530
Q6	0.067	0.282	0.101	0.715	0.085	0.635
Q7	0.083	0.347	0.117	0.737	0.083	0.687
Q8	0.077	0.441	0.120	0.775	0.093	0.712
Q9	0.091	0.566	0.123	0.785	0.093	0.728
Q10	0.112	0.717	**0.131**	0.784	0.112	0.745



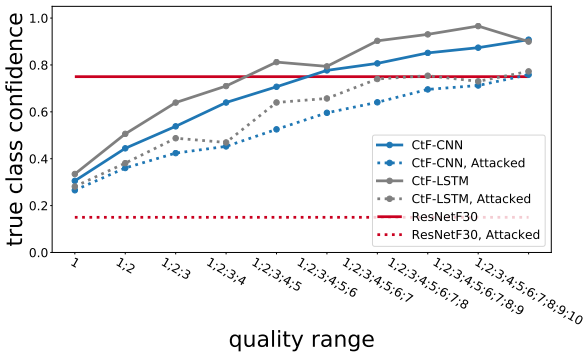
(a) RSCD, Pred. Conf.



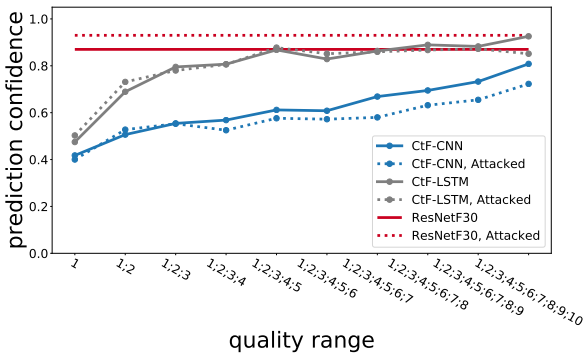
(b) RSCD, True Conf.



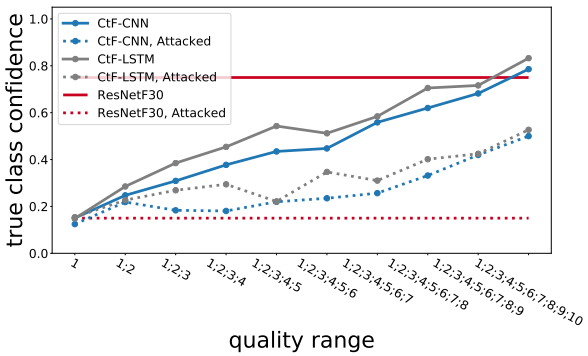
(c) JPEG, Pred. Conf.



(d) JPEG, True Conf.

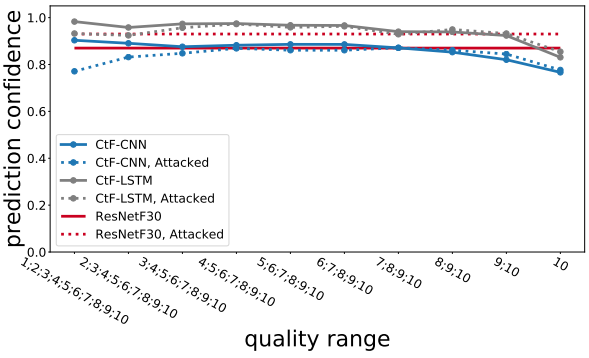


(e) Gaussian, Pred. Conf.

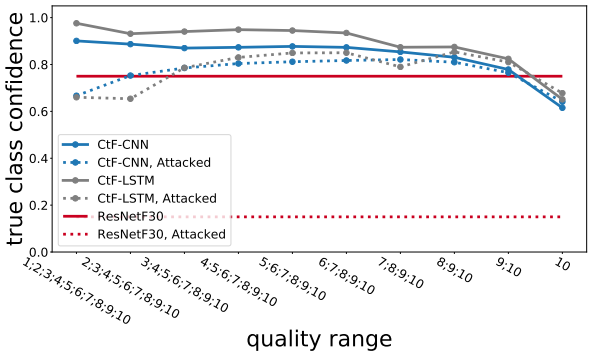


(f) Gaussian, True Conf.

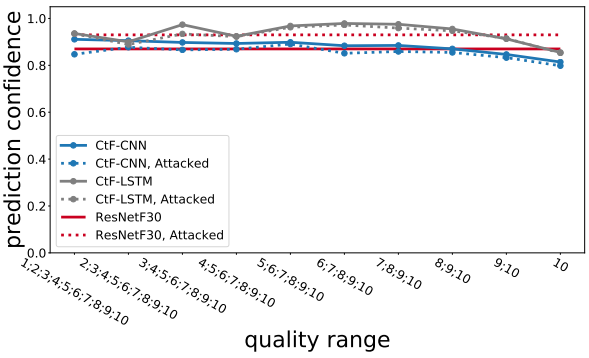
Figure 4. The prediction confidence versus the true-class confidence of the decomposition methods on "gaining" data. These plots show that ResNetF30 is overconfident and incorrect on the attacked images using any of the decomposition methods. Also, we see how applying the CtF framework helps reduce the gap between prediction confidence and true-class confidence, resulting in a more robust visual perception framework. See Figure 6 in the main paper for comparisons of the accuracy values.



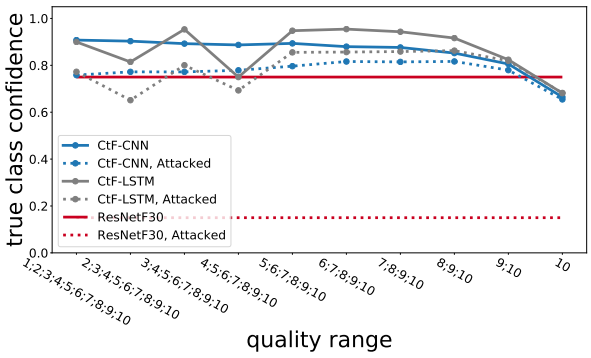
(a) RSCD, Pred. Conf.



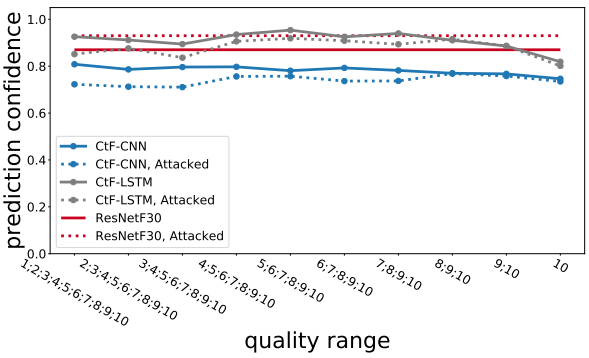
(b) RSCD, True Conf.



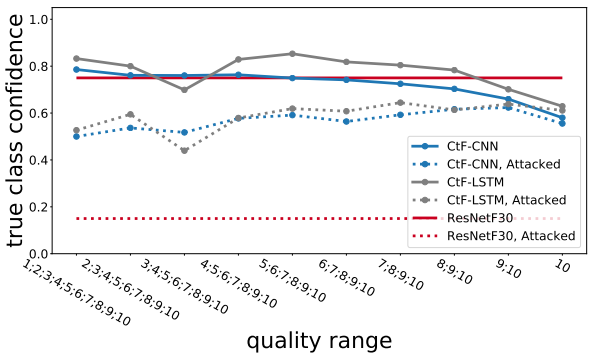
(c) JPEG, Pred. Conf.



(d) JPEG, True Conf.



(e) Gaussian, Pred. Conf.



(f) Gaussian, True Conf.

Figure 5. The prediction confidence versus the true-class confidence of the decomposition methods on "losing" data. Similar to "gaining" data, we see a big gap between ResNetF30's prediction and true-class confidence on the attacked images. The ResNetF30 model is confidently incorrect. We still see the effects of the attacks even when using the CtF framework; however, the gap between the prediction and true-class confidence is relatively minor when using CtF-CNN and CtF-LSTM. See Figure 6 in the main paper for comparisons of the accuracy values.

References

[1] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009. 4

[2] Maureen A Callanan. How parents label objects for young children: The role of input in the acquisition of category hierarchies. *Child Development*, pages 508–523, 1985. 2

[3] François Chollet et al. Keras. https://keras.io, 2015. 6

[4] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018. 4

[5] Christiane Fellbaum. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer, 2010. 2

[6] Daniel Justus, John Brennan, Stephen Bonner, and Andrew Stephen McGough. Predicting the computational cost of deep learning models. In *2018 IEEE international conference on big data (Big Data)*, pages 3873–3882. IEEE, 2018. 4

[7] Edward Kim, Jessica Yarnall, Priya Shah, and Garrett T. Kenyon. A neuromorphic sparse coding defense to adversarial images. In *Proceedings of the International Conference on Neuromorphic Systems*, ICONS '19, New York, NY, USA, 2019. Association for Computing Machinery. 4

[8] Carolyn B. Mervis and Maria A. Crisafi. Order of acquisition of subordinate-, basic-, and superordinate-level categories. *Child Development*, 53(1):258–266, 1982. 2

[9] Tim Pearce, Alexandra Brintrup, and Jun Zhu. Understanding softmax confidence and uncertainty. *arXiv preprint arXiv:2106.04972*, 2021. 6

[10] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019. 2, 3

[11] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 2