Harnessing the Power of Explanations for Incremental Training: A LIME-Based Approach

Arnab Neelim Mazumder*†, Niall Lyons*, Ashutosh Pandey*, Avik Santra*, and Tinoosh Mohsenin†

*Infineon Technologies, Irvine, CA, USA,

†University of Maryland Baltimore County, Baltimore, MD, USA

arnabm1@umbc.edu, niall.lyons@infineon.com, ashutosh.pandey@infineon.com,

avik.santra@infineon.com, and tinoosh@umbc.edu

Abstract—Explainability of neural network prediction is essential to understand feature importance and gain interpretable insight into neural network performance. However, explanations of neural network outcomes are mostly limited to visualization, and there is scarce work that looks to use these explanations as feedback to improve model performance. In this work, model explanations are fed back to the feed-forward training to help the model generalize better. To this extent, a custom weighted loss where the weights are generated by considering the Euclidean distances between true LIME (Local Interpretable Model-Agnostic Explanations) explanations and model-predicted LIME explanations is proposed. Also, in practical training scenarios, developing a solution that can help the model learn sequentially without losing information on previous data distribution is imperative due to the unavailability of all the training data at once. Thus, the framework incorporates the custom weighted loss with Elastic Weight Consolidation (EWC) to maintain performance in sequential testing sets. The proposed custom training procedure results in a consistent enhancement of accuracy ranging from 0.5% to 1.5% throughout all phases of the incremental learning setup compared to traditional loss-based training methods for the keyword spotting task using the Google Speech Commands dataset.

Index Terms—keyword spotting, incremental learning, LIME explanations, elastic weight consolidation, weighted loss, explainable methods

I. INTRODUCTION

Human-Machine interface via voice has become omnipresent in nowadays society. A distinctive feature of voice assistants is that, in order to be used, they first have to be activated by means of a spoken Keyword Spotting (KWS), thereby avoiding computational expenses when it is not required. Thus, KWS can be defined as the task of identification of keywords in audio streams comprising speech and has become a fast-growing technology due to the paradigm shift introduced by deep learning [1]. The earliest approach is based on continuous KWS. One of the advantages of this approach is the flexibility to deal with changing/non-predefined keywords. Whereas the main disadvantage of such KWS systems might reside in the computational complexity dimension and non-availability of all real-world sequential data at once.

Therefore it is usually desired to have Incremental Learning (IL) algorithms, commonly referred to as continual learning [2], [3]. In a continual learning setup, a continuously learning agent at a time step "t" is trained to recognize the tasks $1, .., t-1, t$ while the data $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_{t-1}$, for the tasks

$1, \ldots, t-1$ may or may not be available. Such a learning paradigm has two fundamental trade-offs to overcome. The first of these is Knowledge Transfer (KT), which measures how incremental learning up to task t influences the agent's knowledge about it [2]. In terms of performance, a positive KT suggests that the agent should deliver better accuracy on the task t if allowed to learn it incrementally through tasks $1, \ldots, t-1$ while achieving a low validation error on all of these datasets, assuming that these datasets are seen in the order $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_t$. On the other hand, Semantic Transfer (ST) measures the influence that learning a task t has on the performance of a previous task [4]. A positive ST means that learning a new task t would increase the performance of the model on the previously learned tasks $1, \ldots, t-1$. This compromise between learning a new task and preserving knowledge on previously learned tasks. Multiple methods are proposed in the literature to find the trade-off between KT and ST (architecture-based, memory-based, regularization-based). Here, architectural approaches (e.g., progressive nets [5]) evolve the network size after every task while assimilating the new knowledge with the past knowledge into the new network and memory approaches (e.g., gradient episodic memory [6]) store memory of each of the previous tasks [7] and while learning the new task. In contrast, the regularization method (in this case, elastic weight consolidation) typically assumes a fixed network size and learns a new task while trying to avoid changes to parameters sensitive to previous tasks.

In this work, a regularization-based Elastic Weight Consolidation (EWC) approach is utilized for continual learning where parameter $\boldsymbol{\theta}^*_{1:i-1}$ configuration is achieved at the end of the dataset i, which is expected to solve all the datasets $\mathcal{D}_{1:i}$ [8]. Thus the posterior maximization over the new task is equivalent to likelihood maximization for the new dataset, and the posterior maximization on the previous dataset,

$$\max_{\boldsymbol{\theta}} \log p\left(\boldsymbol{\theta} \mid \mathcal{D}_{1:i}\right) = \max_{\boldsymbol{\theta}} \left[\log p\left(\mathcal{D}_{1:i} \mid \boldsymbol{\theta}\right) + \log p(\boldsymbol{\theta})\right]$$
$$= \max_{\boldsymbol{\theta}} \left[\log p\left(\mathcal{D}_i \mid \boldsymbol{\theta}\right) + \log p\left(\boldsymbol{\theta} \mid \mathcal{D}_{1:i-1}\right)\right] \quad (1)$$

Such an objective can be minimized by adding a regularization loss, which prevents $\boldsymbol{\theta}^*_{1:i}$ from veering too far away from $\boldsymbol{\theta}^*_{1:i-1}$. Since this regularization loss should preserve closeness to the previous solution, the KL-divergence between $p\left(\boldsymbol{\theta} \mid \mathcal{D}_{1:i}\right)$ and $p\left(\boldsymbol{\theta} \mid \mathcal{D}_{1:i-1}\right)$ as the regularization loss is

used. In practice, EWC proposes using the second-order approximation of this KL-divergence:

$$KL\left(p\left(\boldsymbol{\theta} \mid \mathcal{D}_{1:i}\right) \| p\left(\boldsymbol{\theta} \mid \mathcal{D}_{1:i-1}\right)\right) \approx \frac{1}{2} \sum_j F_{jj}\left(\theta_j - \theta_{1:i-1,j}^*\right)^2$$

(2)

Here, F refers to the empirical Fisher matrix, only the diagonal of which is used in the approximation.

As the network is optimized using maximum likelihood estimation, the semantic transfer highly depends on the examples used during training and their similarity coefficient. That is, dataset components with less similarity suffer more semantic loss. To avoid data-dependent optimization, LIME (Local Interpretable Model-Agnostic Explanations) [9], [10] based continual learning where the important semantics are learned using weighted LIME scores in combination with EWC is proposed. The contributions of this paper are:

- A novel and general framework where EWC regularization is combined with model explainability to enhance the classification performance of any neural network is proposed.
- Further, LIME scores of miss-classified samples from the previous task are used as a weighting factor during model optimization to have better semantic transfer learning between tasks in an IL setting and to build more generalized models.

Section II details the proposed approach of LIME-based weighted loss IL with Section III describing the experimental setup and elaborations of the experiments in Section IV.

II. BACKGROUND & PROPOSED FRAMEWORK

The existing literature ([11]) on Explainable AI (XAI) is extensive and focuses particularly on addressing the relationship between model output and its user. Much of this literature is exploratory in nature, as the definition of explainability, intertwined with interpretability, is loosely defined. However, recently some research has been conducted on the use of incorporating XAI and IL to create robust, reliable, non-human-in-the-loop AI systems [12], [13]. The proposed solution utilizes recent XAI approaches introduced in [9], [14] and incorporates them into an IL framework.

The framework shown in Figure 1 aims to bridge this gap and provide a training methodology to enhance the classification accuracy for the KWS task in-midst of an explainable pipeline that can augment IL for the same task. The situation where adding new data to the training regime negatively impacts the learned distribution is known as catastrophic forgetting. In order to prevent this, [8] proposed EWC, which forces the model to retain previous information on top of adding new data. However, although EWC acts as a regularizer to prevent catastrophic forgetting, it also limits the model to learn information from the new data. To address this, we propose using a weighted loss during model retraining where the weights from the samples come from the difference between LIME visuals of the true and predicted classes. As a result, the model will focus more on rectifying the incorrect
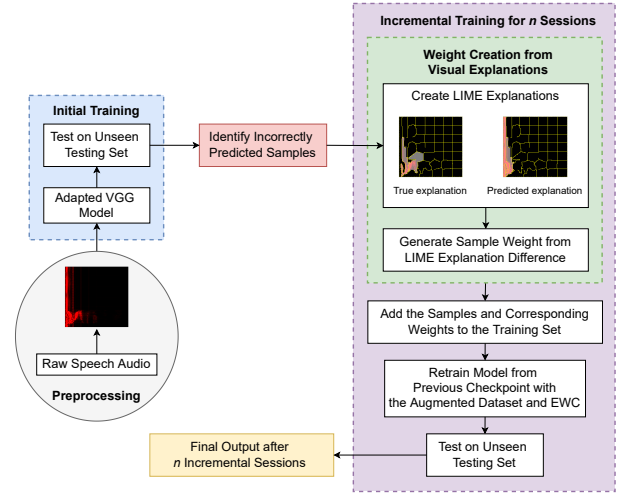

Fig. 1. A high-level overview of the proposed training methodology that integrates explainable weight generation with EWC regularization to maintain model performance in sequential testing sets.

predictions with higher weights, allowing the network to learn the new data better.

A. LIME Visualizations and Feature Scores

Algorithm 1: Feature Score Generation Procedure with LIME

Require: X_{in}, θ^t
$C = Cluster(X_{in})$
$var \leftarrow [null], \sigma = 0.25$
for i in $[1, 2, ...n]$ **do**
   $V = Perturbation(C)$
   $var.append(V)$
end for
$pred = Predict(\theta^t, var)$
$dist = Cosine\ Distance(X_{in}, var)$
$wt = \sqrt{e^{(-dist^2/\sigma^2)}}$
$reg = Linear\ Regression(var, pred, wt)$
$score = \text{coeff}(reg)$

Saliency map-based methods like GradCAM [10], Score-CAM [15], etc., can provide explainable visuals of the heatmap overlaid on top of the original image according to either weight or gradient activation. However, in the context of enhancing the accuracy of the KWS task, determining the importance of segments within a spectrogram to isolate activity regions is highly important. To this end, the LIME-based visual explanations [9] to generate sample weight for the weighted loss is adopted in this work and detailed in Algorithm 1. Initially, the input is segmented with the *slic* [16] clustering algorithm. The clustered input then goes through a perturbation process where the different segments are turned on or off according to the binomial distribution. Each instance of the clustered input thus generated is referred to as a variation. Next, the trained model (for which the explanations are generated) is used to predict the classes of all these

variations. Finally, the scores are created for the segments by fitting a linear regression classifier on the variations and their corresponding predictions, where the cosine distance between the variations and the original input act as weights. Since LIME explanations are based on segmentations created on the spectrogram, it tries to fit a linear classifier for generating an importance score for the segment itself. As a result, LIME provides a qualitative and a quantitative metric to explain the model prediction on an example as shown in Figure 2 (A) and (B).
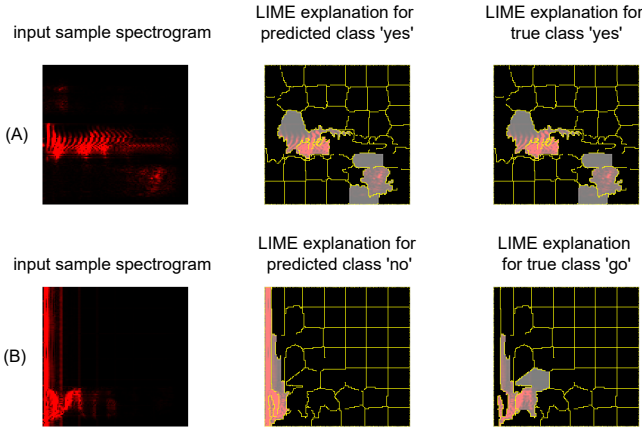

Fig. 2. A qualitative representation of LIME explanations for an accurate classification (A) and a misclassification (B). (A) illustrates the five most influential *slic* clusters (greyed out clusters) for making the prediction 'yes', which coincides with the LIME explanation for the true class. (B) demonstrates the two most influential *slic* clusters for making the prediction 'no', which is different compared to the important *slic* clusters of the true class 'go'.

B. Weighted Loss

In a traditional training setup, all samples are provided a similar influence/weight during loss generation. However, some samples are more difficult to predict than others. More importantly, in an IL setting, it is imperative to learn the incoming data rigorously while preserving the already learned information from the distribution. In this context, using a weighted loss function will force the model to prioritize learning the new samples in back-propagation during the training. For example, if a sample is X_i, with the corresponding model to be $\theta(x_i, w)$ with w parameter weights, and the sample loss to be $L(\hat{y}_i, y)$ then the weighted loss function for N batches in a feed-forward network is given by,

$$Loss = \frac{1}{N} \sum_{i=1}^{N} w_i L(\hat{y}_i, y) \qquad (3)$$

The straightforward way to generate these above-mentioned sample weights would be to assign each incorrectly predicted sample with a high value. However, that process might not represent which samples need more focus than others. To address this, sample weights are generated from the explainable LIME visuals. The weights are considered to be the Euclidean distance between the LIME explanations for the true class (E_t) and the predicted class (E_p) as given by the following:

$$w = \sum_{i=1}^{i=n} (E_p^i - E_t^i)^2 \qquad (4)$$

Here, n denotes the number of segments. The choice behind using Euclidean distance is further justified through Table I, where it is evident that Euclidean distance-based LIME weights outperform Manhattan and Cosine distance-based LIME weights in terms of classification accuracy.

Distance Metric	Accuracy (%)
Euclidean	**90.92 ±0.23**
Manhattan	90.69 ±0.28
Cosine	90.65 ±0.24

C. Elastic Weight Consolidation (EWC)

To address catastrophic forgetting in an IL setting, it is imperative to have some regularization (usually L1 or L2) during training to maintain information from the previous distributions. However, as demonstrated in [8], L1 or L2 normalization most often constrains each weight with the same coefficient, meaning the model can only remember the previous task at the expense of not learning the new one. EWC provides an alternative to this predicament where learning is slowed down on certain weights based on their importance to the previous distribution with a quadratic penalty on the loss, as shown in Equation 5. This scenario further enhances the learning of both tasks and maintains the model performance on each separately. For example, if the loss for the current task is L_{curr}, then EWC regularization is given by the following:

$$Loss = L_{curr} + \sum_{i} \frac{\lambda}{2} F_i (\theta_i - \theta_{A,i}^*)^2 \qquad (5)$$

Here, $\theta_{A,i}$ are the parameters from the previous task, and θ_i are the parameters from the current task. F_i represents the parameters of the previous task's Fisher Information Matrix (FIM). λ controls the amount of EWC regularization applied to the current loss. A lower λ value favors learning the new task more at the expense of forgetting the previous task and vice versa.

III. EXPERIMENTAL SETUP

The Google Speech Commands dataset [17] forms the basis of our experimental protocol. The process starts with normalizing all audio files to 16,000 samples through zero-padding. Following this, we generate spectrograms from these files, thereby considerably diminishing the computational load for subsequent deployment of Deep Neural Networks (DNNs). The derived spectrograms further facilitate LIME-based explanations that serve as valuable feedback in the learning mechanism. For this study, a VGG-like architecture is leveraged, albeit with only four blocks and lesser filters across all layers compared to the conventional VGG structure. This modified

Algorithm 2: Incremental Training Procedure with LIME-based Weighted Loss and EWC.

Require: D^m, D^v, D^t
Initial Training :
1: Initialize model θ^0
2: $\theta^m = \text{train}\left(D^m, \theta^0, D^v\right)$
3: $Acc^m = \text{eval}\left(\theta^m, D^t\right)$
Generate LIME weights
4: **for** D^{v_i} in $[D^{v_1}, D^{v_2}, ...D^{v_n}]$ **do**
5: $\quad D^{I_i} = \{(d,y) \in D^{v_i}|pred(\theta^m, d) \neq y\}$
6: $\quad LIME\_pred^{I_i} = lime_p(\theta^m, D^{I_i})$
7: $\quad LIME\_true^{I_i} = lime_t(\theta^m, D^{I_i})$
8: $\quad w^{I_i} = l\_wt(LIME\_pred^{I_i}, LIME\_true^{I_i})$
9: **end for**
Incremental Training :
10: Initialize $D = D^m, \theta^0 = \theta^m, w = 1$
11: **for** D^{I_i} in $[D^{I_1}, D^{v_2}, ...D^{I_n}]$ **do**
12: $\quad$ Add new data: $D_{new} = D \cup D^{I_i}, w = w \cup w^{I_i}$
13: $\quad FIM = gen\_fim(\theta^{i-1}, 5\% \text{ of } D_{new})$
14: $\quad \theta^i = EWC\_trn(\theta^{i-1}, D_{new}, w, D^{v_i}, FIM)$
15: $\quad Acc^i = eval(\theta^i, D^t)$
16: **end for**

VGG structure is henceforth referenced as the study's adapted VGG architecture (illustrated in Equation 6). In this notation, $2 \times (8\,Conv_{3\times3})$ denotes 2 consecutive convolutional layers with 8 filters and a 3×3 kernel shape, $32\,pool_{2\times2}$ signifies a 2×2 maxpooling layer with 32 filters, and $1000\,FC$ represents a fully-connected layer hosting 1000 neurons.

$$(in) - 2 \times (8\,Conv_{3\times3}) - 2 \times (16\,Conv_{3\times3}) - 3 \times (32\,Conv_{3\times3}) -$$
$$(32\,pool_{2\times2}) - 3 \times (64\,Conv_{3\times3}) - (64\,pool_{2\times2}) - (1000\,FC) - (out) \quad (6)$$

The comprehensive training approach is depicted in Figure 1 and elaborated in Algorithm 2. To ensure viable sequential retraining, we partition the data into training, validation, and test sets at 80%, 10%, and 10% respectively, strictly adhering to speaker information for the division. This ensures complete containment of a speaker's utterances within a single set, thus preventing any potential leakage of samples between sets. For the purpose of retraining, misclassified samples from the validation sets are moved into the training set. These samples are weighted based on the Euclidean distance between the LIME explanation of the predicted class ($lime_p$) and the actual class ($lime_t$), as denoted in line 8 of Algorithm 2. The stepwise training procedure then unfolds by introducing these misclassified samples and corresponding LIME weights into the training data D_m. This incremental methodology continues for all sequential training iterations. The choice of this strategy arises from the notion that the user may not always have access to previously unseen test data or validation data. It also offers the advantage of significant reduction in training time, given that only the misclassified samples are considered for retraining, instead of the entire validation set. To formulate the Fisher Information Matrix (FIM) during EWC-based training, we randomly select 5% of the correctly predicted samples

from the expanded training set in each session. In all sessions, the retraining commences utilizing the weights inherited from the previous session, leveraging the Adam optimizer (with a learning rate of 0.001) and a batch size of 512.

IV. RESULTS & DISCUSSION

We perform six separate runs on the complete dataset for weighted loss training and regular loss training to create an analogy between the two methods. The weighted loss method results in 90.9% Top-1 accuracy, which is around 1% better than regular loss-based training (90.2%). Figure 3 shows the performance of the first ten classes during testing in the form of confusion matrices. The weighted loss training method adopted in this work performs better and gets more predictions right in most classes than traditional loss-based training, proving that our custom loss enhances performance. To further accommodate EWC according to Equation 5, we aim to choose the optimal parameter (λ), which controls the influence of EWC regularization while generating the loss. A higher λ value allows aggressive learning of the newer task with the caveat of forgetting the older ones. Hence we experiment with six identical EWC and weighted loss-based IL setups for six different λ values. As per the line graph shown in Figure 4, it is apparent that $\lambda = 1$ provides us with the most stable performance in terms of test set accuracy throughout all six sessions of IL.

TABLE II
STANDARD ERROR COMPARISON AMONG TRADITIONAL LOSS, WEIGHTED LOSS AND WEIGHTED LOSS WITH EWC FOR 16 STAGE INCREMENTAL LEARNING OVER THREE SEPARATE RUNS

# of Sess.	Trad. Loss	W_Loss	W_Loss + EWC
1	89.37 ±0.75	90.17 ±0.71	90.43 ±1.08
4	89.90 ±0.56	90.20 ±0.78	91.00 ±0.95
8	89.73 ±0.51	90.13 ±0.50	**91.07 ±0.49**
12	89.93 ±0.75	90.03 ±0.51	90.63 ±0.70
16	89.57 ±0.60	90.03 ±0.46	**90.87 ±0.65**

Table II compares the performance of the EWC-based weighted loss IL to traditional IL and stand-alone weighted loss IL, respectively. The 16-stage IL experiment suggests that the weighted loss incremental training performs better than traditional incremental training throughout all the sessions. And, when coupled with EWC regularization, the accuracy is better for EWC-based weighted loss IL than the stand-alone weighted loss IL, further providing evidence that our proposed method retains previous information better than traditional IL setups. Moreover, we see the test accuracy follows a reducing trend for all three methods as more incremental stages are trained. This effect is the result of some information from the earlier stages being lost. Finally, our proposed method has an improvement of at least 0.5-1.5% over traditional loss based IL setup for all stages of IL and even with higher training stages our method maintains better performance compared to both traditional and weighted loss based IL setups.
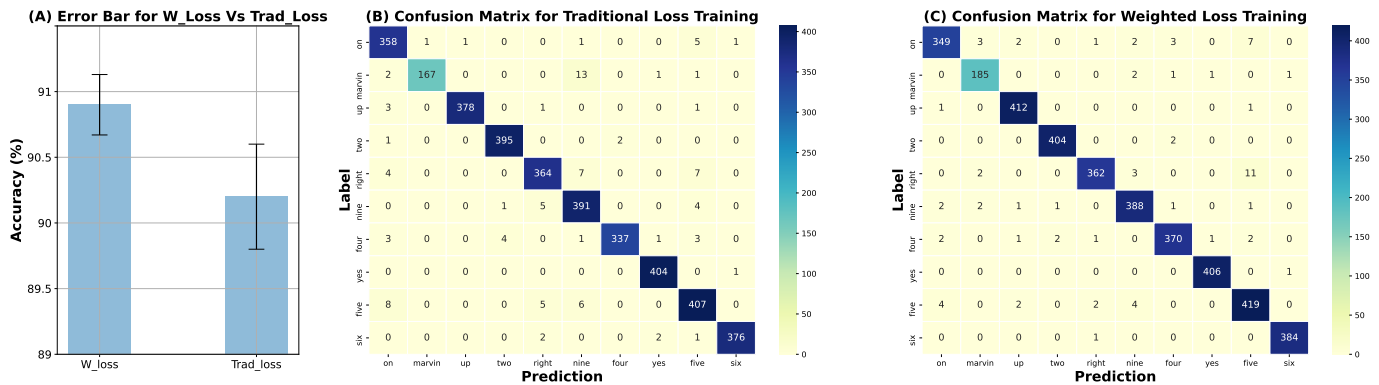
Fig. 3. (A), (B), and (C) correspond to the error bar for two training methods over six runs, confusion matrix for weighted loss-based training, and traditional loss-based training, respectively, on the complete dataset. Only the first ten classes are represented in the confusion matrix figures. The rest of the classes follow a similar trend as well.
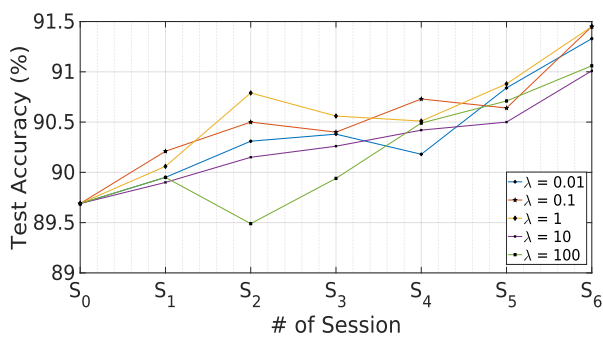

Fig. 4. The experimental insight allows us to conclude that $\lambda = 1$ provides the best performance in terms of accuracy throughout the six different sessions of IL.

V. CONCLUSION

This work aims to take advantage of the LIME visuals of a model prediction and use these explainable insights to improve the performance of DNNs. By differentiating the explanations for actual and predicted classes, proportional weights are generated for false predictions and used during the consequent IL stage retraining. On top of it, the enhancement brought upon by EWC regularization to maintain neural network performance across IL scenarios is explored. The experiments suggest that the weighted loss training coupled with EWC provides at least 0.5-1.5% improvement on all IL stages over the traditional loss-based IL. Our future work will expand the scope of this methodology to include weights from explainable visuals in the context of metric learning to further bolster the classification network's performance.

REFERENCES

[1] I. López-Espejo, Z.-H. Tan, J. Hansen, and J. Jensen, "Deep spoken keyword spotting: An overview," 2021. [Online]. Available: https://arxiv.org/abs/2111.10592

[2] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro, "Learning to learn without forgetting by maximizing transfer and minimizing interference," 2018. [Online]. Available: https://arxiv.org/abs/1810.11910

[3] S. Jha, M. Schiemer, and J. Ye, "Continual learning in human activity recognition: an empirical analysis of regularization," 2020. [Online]. Available: https://arxiv.org/abs/2007.03032

[4] V. V. Ramasesh, E. Dyer, and M. Raghu, "Anatomy of catastrophic forgetting: Hidden representations and task semantics," 2020. [Online]. Available: https://arxiv.org/abs/2007.07400

[5] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," 2016. [Online]. Available: https://arxiv.org/abs/1606.04671

[6] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," 2017. [Online]. Available: https://arxiv.org/abs/1706.08840

[7] J. Knoblauch, H. Husain, and T. Diethe, "Optimal continual learning has perfect memory and is np-hard," 2020. [Online]. Available: https://arxiv.org/abs/2006.05188

[8] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[9] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[10] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that?" *arXiv preprint arXiv:1611.07450*, 2016.

[11] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery." *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[12] S. Ede, S. Baghdadlian, L. Weber, A. Nguyen, D. Zanca, W. Samek, and S. Lapuschkin, "Explain to not forget: Defending against catastrophic forgetting with xai," in *Machine Learning and Knowledge Extraction: 6th IFIP TC 5, TC 12, WG 8.4, WG 8.9, WG 12.9 International Cross-Domain Conference, CD-MAKE 2022, Vienna, Austria, August 23–26, 2022, Proceedings*. Springer, 2022, pp. 1–18.

[13] L. Weber, S. Lapuschkin, A. Binder, and W. Samek, "Beyond explaining: Opportunities and challenges of xai-based model improvement," *Information Fusion*, 2022.

[14] H. Sun, L. Servadei, H. Feng, M. Stephan, R. Wille, and A. Santra, "Utilizing explainable ai for improving the performance of neural networks," 2022. [Online]. Available: https://arxiv.org/abs/2210.04686

[15] H. Wang, Z. Wang, M. Du, F. Yang, Z. Zhang, S. Ding, P. Mardziel, and X. Hu, "Score-cam: Score-weighted visual explanations for convolutional neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 24–25.

[16] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," Tech. Rep., 2010.

[17] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.