Supplementary Material for: Psycholinguistics meets Continual Learning: Measuring Catastrophic Forgetting in Visual Question Answering

Claudio Greco[1]
claudio.greco@unitn.it

Barbara Plank[2]
bplank@itu.dk

Raquel Fernández[3]
raquel.fernandez@uva.nl

Raffaella Bernardi[1]
raffaella.bernardi@unitn.it

[1]University of Trento [2]IT University of Copenhagen [3]University of Amsterdam

1 Implementation details

All models were trained using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.0005 and with a batch size of 64. We stopped the training of the models whenever their accuracy on the validation set did not increase for 3 times in a row. Word embeddings had a size of 300. RNNs had two hidden layers and LSTM cells had a size of 1024. MLPs had one hidden layer of size 1024. We used the implementation released by (Johnson et al., 2017) for the LSTM+CNN+SA architecture.

2 Hyperparameter search

For *EWC*, we searched for the best λ value among 100, 1000, 10000. For *Rehearsal*, we considered sampling size values of 100, 1000, 10000 training examples from Task A. We reported results for the models having the highest *CL score* computed according to the validation sets of both tasks. For *EWC*, the best model had $\lambda = 100$; for *Rehearsal*, the best model used 10000 training examples from Task A in both orders, WH→Y/N and Y/N→WH.

3 Continual Learning Evaluation Measures

Besides standard *Accuracy* (*Acc*), we consider metrics that have been introduced specifically to evaluate continual learning. In general, there is not much agreement among authors about the best metrics to evaluate continual learning models. Thus, Díaz-Rodríguez et al. (2018) propose a set of comprehensive metrics which allow to evaluate different factors of continual learning models, such as accuracy, forgetting, backward/forward knowledge transfer, memory overhead, and computational efficiency. In this paper, we focus on evaluating accuracy and forgetting across tasks. First, the authors define a measure describing the overall behavior of continual learning models. In particular, for each measure i describing a particular aspect of a model, let c_i (where $c_i \in [0, 1]$) be its average value and s_i (where $s_i \in [0, 1]$) be its standard deviation across r runs. Let $w_i \in [0, 1]$ (where $\sum_i^{\mathcal{C}} w_i = 1$) be the weight given to measure i. Then, the *CL score*, which measures the overall score of the model across tasks, is defined. Higher values are better and the measure lies in the range $[0, 1]$. Formally, it is computed as follows:

$$\textit{CL score} = \sum_{i=1}^{|C|} w_i c_i$$

Let $R \in \mathbb{R}^{N \times N}$ be the train-test accuracy matrix, whose element $R_{i,j}$ is equal to the test accuracy on task j after having trained the model up to task i, where N is the number of tasks. In the evaluation of the *CL score*, we take the following measures into account:

- *Mean accuracy* (*Mean acc*) (Díaz-Rodríguez et al., 2018), which measures the overall accuracy of the model on the learned tasks. Higher values are better and the measure lies in the range $[0, 1]$. Formally, it is defined as:

$$\textit{Mean acc} = \frac{\sum_{i \geq j}^{N} R_{i,j}}{\frac{N(N+1)}{2}}$$

- *Remembering* (*Rem*) (Díaz-Rodríguez et al., 2018), which measures how much the model remembers how to perform previously learned tasks. Higher values are better and the measure lies in the range $[0, 1]$. Formally, it is defined as:

$$\textit{Rem} = 1 - |min(BWT, 0)|,$$

where *Backward transfer* (*BWT*) allows to measure the influence that learning a task has

on the performance of the previously learned tasks and it is formally defined as:

$$BWT = \frac{N_{i=2}^{N} \sum_{j=1}^{i-1} (R_{i,j} - R_{j,j})}{\frac{N(N-1)}{2}}$$

- *Intransigence* (*Int*) (Chaudhry et al., 2018), which captures how much a model is regularized towards preserving past knowledge and as a consequence less capable of learning new tasks. Lower values are better and the measure lies in the range $[-1, 1]$. Formally, intransigence on the k-th task is defined as:

$$I_k = a_k^* - a_{k,k},$$

where $a_{k,k}$ denotes the accuracy on task k of the model trained sequentially up to task k and a_k^* denotes the accuracy on task k of the *Cumulative* model trained on tasks $1, \ldots, k$. In this paper, we only measure intransigence for the second task, because we take only two tasks into account and it does not make sense to compute intransigence for the first task. Hence, Int denotes I_2.

CL score requires that each measure lies in the range $[0, 1]$ and that higher values are better. *Mean acc* and *Rem* already satisfy these constraints, whereas *Int* does not. Hence, when computing *CL score* in the case of *Int*, c_i is transformed to $c_i = 1 - (c_i + 1)/2$ to scale its range to $[0, 1]$ and to preserve the monotonicity of *CL score*.

4 Elastic Weight Consolidation

Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) is a regularization approach which introduces plasticity in artificial neural networks by slowing down learning in weights which are important to solve previously learned tasks. The method takes inspiration from the human brain, in which the plasticity of synapses which are important to solve previously learned tasks is reduced. *EWC* adds a regularization term to the loss function allowing the model to converge to parameters where it has a low error for both tasks. In particular, if Task A and Task B have to be learned sequentially *EWC*, after having learned Task A, computes the Fisher Information Matrix, whose i-th diagonal element assesses how important parameter i of the model is to solve Task A. Then, the model is trained on Task B starting from the parameters previously learned to solve Task A by minimizing the following loss function:

$$L = L_B(\theta) + \frac{\lambda}{2} \sum_i F_{i,i} (\theta_i - \theta_i^A)^2,$$

where L_B is the loss function of Task B, $F_{i,i}$ is the i-th diagonal element of the Fisher Information Matrix, θ_i is the i-th parameter, θ_i^A is the optimal i-th parameter for Task A, and λ controls the regularization strength, i.e. the higher it is, the more it is important to remember Task A.

5 Confusion matrices

Tables 1 to 5 show the confusion matrices of the Wh, *Naive*, *Cumulative*, *Rehearsal*, and *EWC* models, respectively, on the WH $\rightarrow$ Y/N setup. Tables 1 and 7 to 10, instead, show the confusion matrices of the Y/N, *Naive*, *Cumulative*, *Rehearsal*, and *EWC* models, respectively, on the Y/N $\rightarrow$ WH setup. In particular, predictions on these confusion matrices are grouped according to their category, so that rows represent the question type each question belongs to, columns represent the category each answer belongs to, and cells show the number of predictions the model obtains for a particular question type and answer category.

6 Neuron activations

Figures 1 and 2 show the neuron activations on the penultimate hidden layer of *Naive* model for the I) WH $\rightarrow$ Y/N setup and the model trained independently on Y/N-q, respectively. All the visualizations of neuron activations reported in the paper are obtained by computing the vectors containing the neuron activations of the penultimate hidden layer of the model during forward propagation and by plotting the resulting vectors transformed into two dimensions through *t-distributed Stochastic Neighbor Embedding (t-SNE)* (Maaten and Hinton, 2008).

Wh	query_color	query_shape	query_size	query_material	Yes/No
query_color	6752	0	0	0	0
query_shape	0	6702	0	0	0
query_size	0	0	6666	0	0
query_material	0	0	0	6653	0
equal_color	1204	14	2088	3	0
equal_shape	26	1150	2232	2	0
equal_size	0	0	3430	0	0
equal_material	21	34	1440	2037	0

Table 1: Confusion matrix of the model trained independently on Wh-q.

Naive	query_color	query_shape	query_size	query_material	Yes/No
query_color	15	0	0	0	6738
query_shape	0	81	0	0	6621
query_size	0	0	0	0	6666
query_material	0	0	0	148	6505
equal_color	0	0	0	0	3309
equal_shape	0	0	0	0	3410
equal_size	0	0	0	0	3430
equal_material	0	0	0	0	3532

Table 2: Confusion matrix of the *Naive* model on the WH $\rightarrow$ Y/N setup.

Cumulative	query_color	query_shape	query_size	query_material	Yes/No
query_color	6752	0	0	1	0
query_shape	0	6702	0	0	0
query_size	0	0	6665	1	0
query_material	0	0	0	6653	0
equal_color	0	0	0	0	3309
equal_shape	0	0	0	0	3410
equal_size	0	0	0	0	3430
equal_material	0	0	0	0	3532

Table 3: Confusion matrix of the *Cumulative* model on the WH $\rightarrow$ Y/N setup.

Rehearsal	query_color	query_shape	query_size	query_material	Yes/No
query_color	6743	1	8	1	0
query_shape	0	6702	0	0	0
query_size	0	0	6664	0	2
query_material	1	0	1	6651	0
equal_color	0	0	0	0	3309
equal_shape	0	0	0	0	3410
equal_size	0	0	0	0	3430
equal_material	0	0	0	0	3532

Table 4: Confusion matrix of the best *Rehearsal* model on the WH $\rightarrow$ Y/N setup.

EWC	query_color	query_shape	query_size	query_material	Yes/No
query_color	6715	0	0	1	37
query_shape	0	5479	0	0	1223
query_size	0	0	0	0	6657
query_material	0	0	0	1337	5316
equal_color	0	0	0	0	3309
equal_shape	0	2	0	0	3408
equal_size	0	0	1	0	3429
equal_material	0	0	0	0	3532

Table 5: Confusion matrix of the best *EWC* model on the WH → Y/N setup.

Y/N	query_color	query_shape	query_size	query_material	Yes/No
query_color	0	0	0	0	6753
query_shape	0	0	0	0	6753
query_size	0	0	0	0	6666
query_material	0	0	0	0	6653
equal_color	0	0	0	0	3309
equal_shape	0	0	0	0	3410
equal_size	0	0	0	0	3430
equal_material	0	0	0	0	3532

Table 6: Confusion matrix of the model trained independently on Y/N-q.

Naive	query_color	query_shape	query_size	query_material	Yes/No
query_color	6753	0	0	0	0
query_shape	0	6701	1	0	0
query_size	0	0	6666	0	0
query_material	1	0	1	6651	0
equal_color	2732	38	229	310	0
equal_shape	1317	1144	346	603	0
equal_size	1330	16	1559	525	0
equal_material	1297	0	30	2205	0

Table 7: Confusion matrix of the *Naive* model on the Y/N → WH setup.

Cumulative	query_color	query_shape	query_size	query_material	Yes/No
query_color	6753	0	0	0	0
query_shape	1	6701	0	0	0
query_size	0	0	6666	0	0
query_material	0	0	0	6653	0
equal_color	0	0	0	0	3309
equal_shape	0	0	0	0	3410
equal_size	0	0	0	0	3430
equal_material	0	0	0	0	3532

Table 8: Confusion matrix of the *Cumulative* model on the Y/N → WH setup.

Rehearsal	query_color	query_shape	query_size	query_material	Yes/No
query_color	6752	0	1	0	0
query_shape	0	6702	0	0	0
query_size	0	0	6666	0	0
query_material	1	0	1	6651	0
equal_color	0	0	0	0	3309
equal_shape	1	0	0	0	3409
equal_size	0	0	1	0	3429
equal_material	0	0	0	0	3532

Table 9: Confusion matrix of the best *Rehearsal* model on the Y/N → WH setup.

EWC	query_color	query_shape	query_size	query_material	Yes/No
query_color	6748	4	0	1	0
query_shape	0	6701	1	0	0
query_size	0	0	6666	0	0
query_material	1	0	0	6652	0
equal_color	3110	9	17	173	0
equal_shape	801	1214	69	1326	0
equal_size	542	35	35	1674	2
equal_material	464	2	1	3065	0

Table 10: Confusion matrix of the best *EWC* model on the Y/N → WH setup.

Figure 1: Analysis of the neuron activations on the penultimate hidden layer of the *Naive* model for the I) WH → Y/N setup.

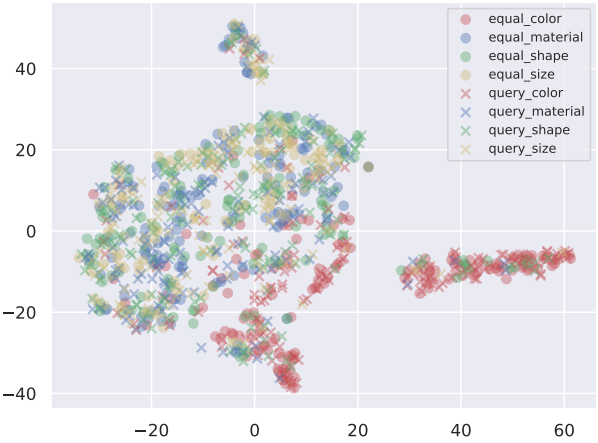

Figure 2: Analysis of the neuron activations on the penultimate hidden layer of the model trained independently on Y/N-q.

References

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip Torr. 2018. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*.

Natalia Díaz-Rodríguez, Vincenzo Lomonaco, David Filliat, and Davide Maltoni. 2018. Don't forget, there is more than forgetting: new metrics for continual learning. In *Workshop on Continual Learning, NeurIPS*.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. Inferring and executing programs for visual reasoning. In *ICCV*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *PNAS*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.