Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)

Revisiting Neural Networks for Continual Learning: An Architectural Perspective

Aojun Lu[1] , **Tao Feng**[2] , **Hangjie Yuan**[3] , **Xiaotian Song**[1] and **Yanan Sun**[1*]
[1]Sichuan University
[2]Tsinghua University
[3]Zhejiang University
aojunlu@stu.scu.edu.cn, fengtao.hi@gmail.com, hj.yuan@zju.edu.cn
songxt@stu.scu.edu.cn, ysun@scu.edu.cn

SIN5006 - INTELIGÊNCIA COMPUTACIONAL

PROPOSTA PARA TRABALHO SEMESTRAL

NILTON TADASHI ENTA - 12730911
VITOR CAMARGO PINHEIRO - 12693156

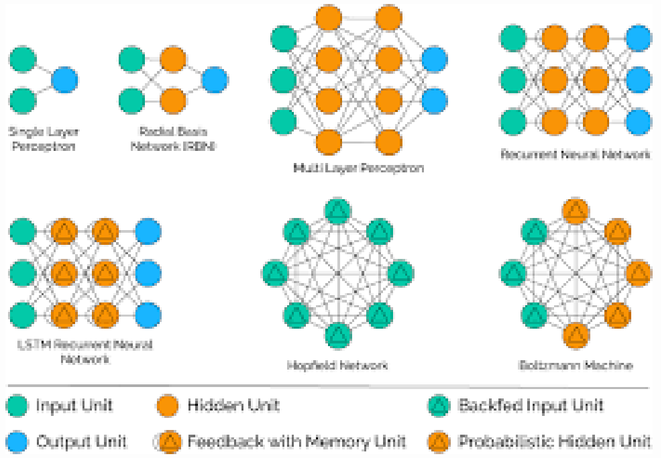
https://arxiv.org/pdf/2404.14829

REDE NEURAL(NN) - ARQUITETURAS

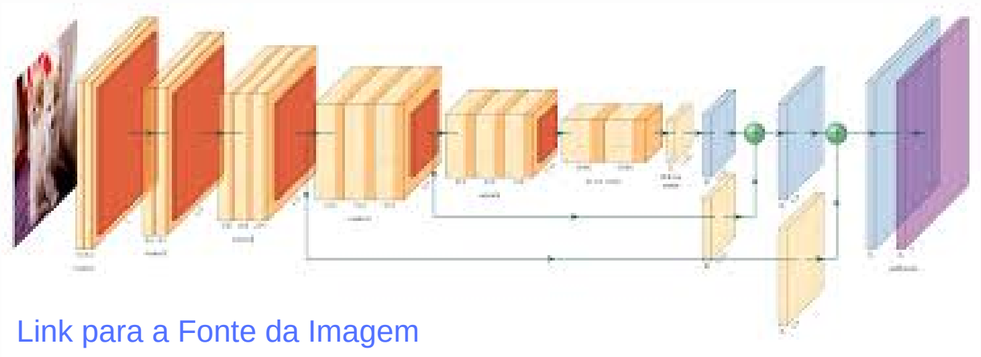
A Arquitetura de uma Rede Neural, ou seja, seus componentes são formados por:

- Camadas de entrada, ocultas e saída
- Neurônios
- Funções de ativação
- Conexões e pesos

Exemplos: MLP, CNN, RNN, Transformers entre outros.



Link para a Fonte da Imagem



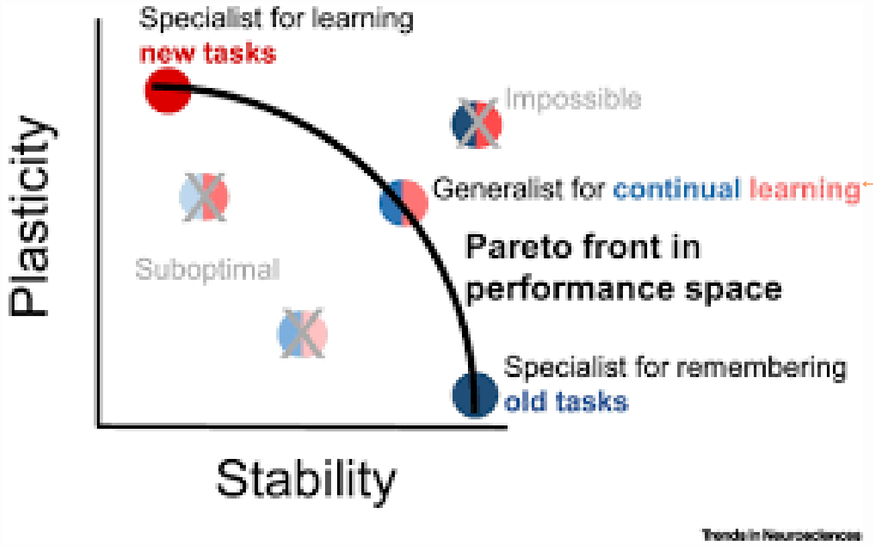
Link para a Fonte da Imagem

O artigo propõe uma abordagem arquitetural para a otimização de aprendizado contínuo em redes neurais para reconhecimento de imagem.

DILEMA DO APRENDIZADO CONTÍNUO (CL) EM NN

Um modelo pode ser treinado uma única vez com todos os dados disponíveis.

O **Aprendizado Contínuo** (*Continuous Learning)* é quando esse modelo recebe novos dados e precisa manter o conhecimento adquirido anteriormente. Quando esse aprendizado não é mantido, chama-se de ***Catastrophic Forgetting***.



Fonte da imagem

Trade-off entre **Estabilidade** e **Plasticidade**

O dilema existente ao inserir novos dados é manter a estabilidade do modelo e ao mesmo tempo preservar sua capacidade de expandir seu aprendizado a novas classes.

Estabilidade *(Stability) refere-se a capacidade do modelo de não perder o aprendizado já adquirido com o incremento de novos dados.*

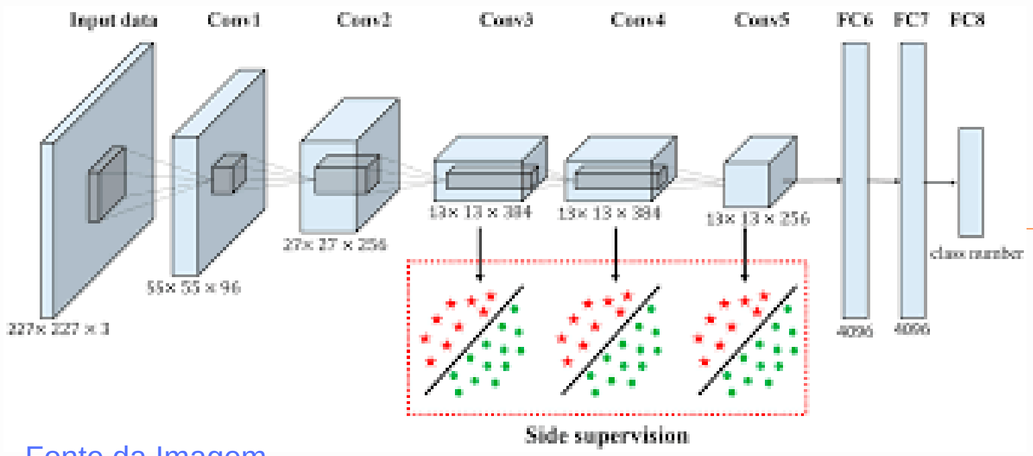
Plasticidade *(Plasticity) refere-se a capacidade do modelo aprender com esses novos dados.*

ALEXNET

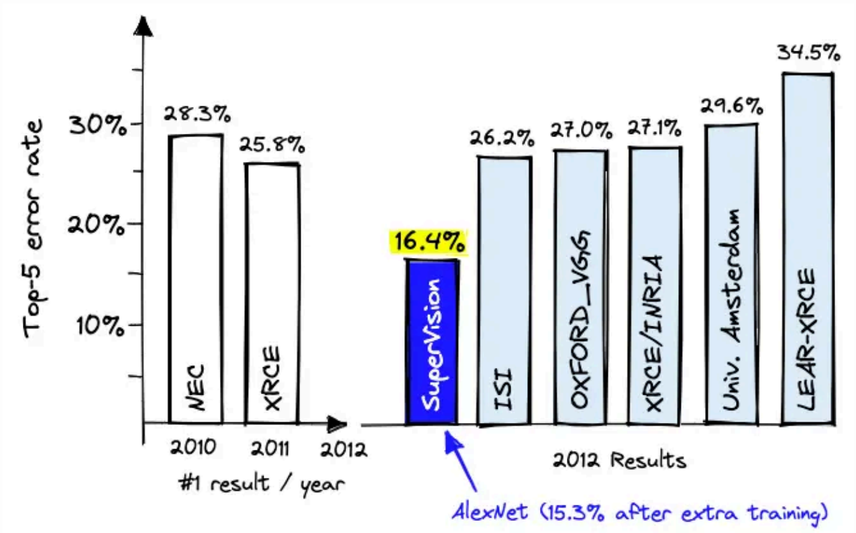
AlexNet é uma rede neural convolucional que revolucionou o campo do aprendizado profundo ao vencer a competição **ImageNet Large Scale Visual Recognition Challenge 2012**.

Reduziu o **erros de classificação de imagens do ImageNet de 26% para 16%,** um avanço gigante na época.

Inspirou redes mais profundas, como VGG, **ResNet** e EfficientNet.



Fonte da Imagem



Arquitetura de 8 camadas:

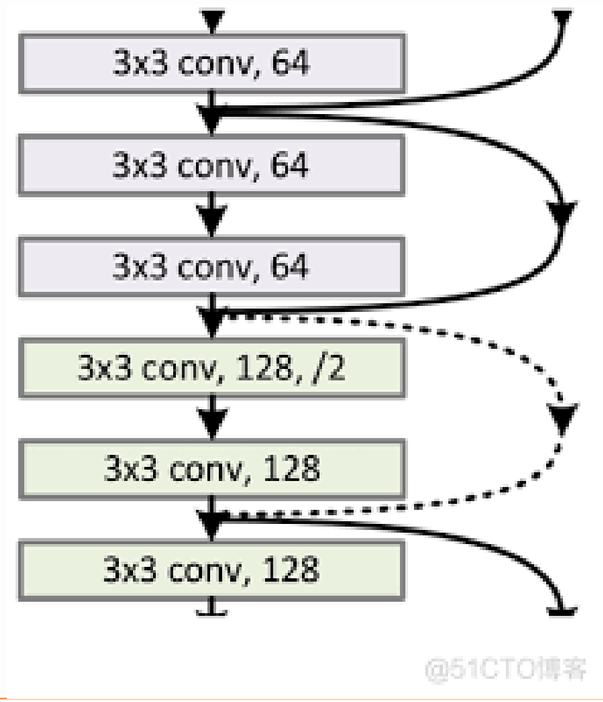
- ***Conv1:*** *96 filtros de tamanho 11×11, stride 4. ReLU e MaxPooling (3×3, stride 2).*
- ***Conv2:*** *256 filtros de 5×5, stride 1. ReLU e MaxPooling (3×3, stride 2).*
- ***Conv3 e Conv4:*** *384 filtros de 3×3, stride 1. ReLU.*
- ***Conv5:*** *256 filtros de 3×3, stride 1. ReLU e MaxPooling (3×3, stride 2).*
- ***Fully Conected 1:*** *4096 neurônios + ReLU + Dropout.*
- ***Fully Conected 2:*** *4096 neurônios + ReLU + Dropout.*
- ***Fully Conected 3****: Softmax n neurônios para classificação.*

RESTNET

ResNet (*Residual Network*) é uma arquitetura de Rede Neural Convolucional (*Convolutional Neural Network*) criada por pesquisadores da Microsoft em 2015.

Foi um dos maiores avanços em aprendizado profundo, pois **resolveu o problema do *vanishing gradient***.

- Atalhos Residuais (*Skip Connections*)
- Se uma camada não aprender nada útil, o atalho garante que pelo menos a entrada original seja passada adiante



Variantes da ResNet

- ***ResNet-18:*** *18 camadas*
- ***ResNet-34:*** *34 camadas*
- ***ResNet-50:*** *50 camadas (introduz blocos Bottleneck)*
- ***ResNet-101:*** *101 camadas*
- ***ResNet-152****: 152 camadas*

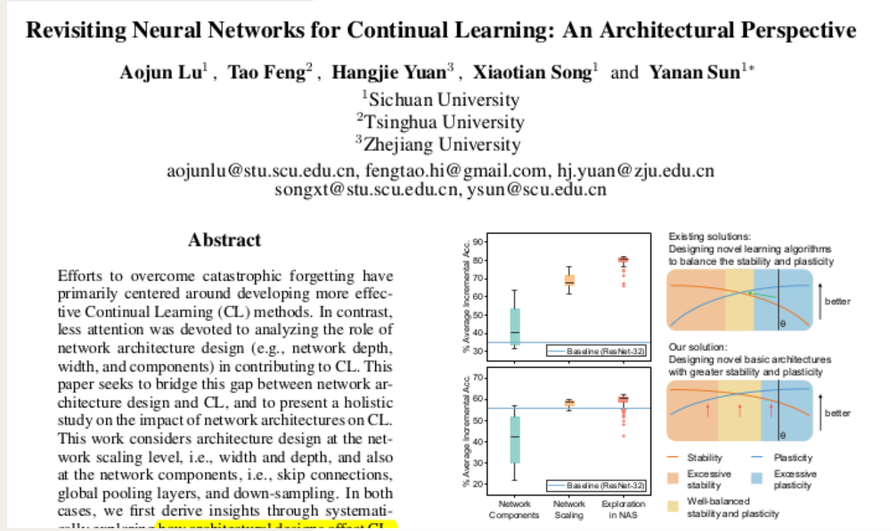
O ARTIGO

O artigo explora uma abordagem arquitetural para aprimorar uma solução para o desafio do Aprendizado Contínuo. Diferente de abordagens como a de **regularização** e **atualização de parâmetros**, como citado no trecho a seguir, retirados do artigo:

*"In contrast, less attention was devoted **to analyzing the role of network architecture design (e.g., network depth, width, and components)** in contributing to CL. This paper seeks to bridge this gap between network architecture design and CL..."*

O autor explora a seleção de parâmetros citados acima e a criação de um ***search space*** customizado para CL a partir de um algoritmo evolutivo.

.



Revisiting Neural Networks for Continual Learning: An Architectural Perspective

Aojun Lu[1], **Tao Feng**[2], **Hangjie Yuan**[3], **Xiaotian Song**[1] and **Yanan Sun**[1*]

[1]Sichuan University
[2]Tsinghua University
[3]Zhejiang University

aojunlu@stu.scu.edu.cn, fengtao.hi@gmail.com, hj.yuan@zju.edu.cn
songxt@stu.scu.edu.cn, ysun@scu.edu.cn

Abstract

Efforts to overcome catastrophic forgetting have primarily centered around developing more effective Continual Learning (CL) methods. In contrast, less attention was devoted to analyzing the role of network architecture design (e.g., network depth, width, and components) in contributing to CL. This paper seeks to bridge this gap between network architecture design and CL, and to present a holistic study on the impact of network architectures on CL. This work considers architecture design at the network scaling level, i.e., width and depth, and also at the network components, i.e., skip connections, global pooling layers, and down-sampling. In both cases, we first derive insights through systemati-

O ARTIGO

Há duas formas de selecionar a melhor arquitetura de uma rede neural:

- **Hand Craft** é uma abordagem manual com um consumo de tempo e esforço humano com resultados variáveis (*abordagem selecionada*)

- **Mecanismos de seleção automatizados** como **NAS** (*Neural Architecture Search*) que ainda dependem do trabalho humano para preparar o ***search space***.

JUSTIFICATIVA DA ESCOLHA

A escolha do artigo apresentado deu-se por alguns motivos, como:

- Interesse por CNNs e *Continuous Learning*.
- Interesse em compreender, na prática, os resultados obtidos pelos autores e as melhorias apresentadas.
- Testar a solução do artigo, com intuito de obter novos resultados em determinados contextos.

TODO MATERIAL NECESSÁRIO PARA O EXPERIMENTO ENCONTRA-SE DISPONÍVEL EM:

byyx666/**ArchCraft**

This is official code implementation of the

***Dataset* disponível em torchvision[1]**

```
from torchvision import datasets

df=datasets.CIFAR100(
                        path,
                        train=True,
                        download=True,
                        ...
                    )
```

1. torchvision é uma biblioteca que faz parte do projeto PyTorch. Confira a documentação da biblioteca em: https://pytorch.org/vision/stable/index.html

SOLUÇÃO PROPOSTA - ARCHCRAFT

SELEÇÃO DE HIPERPARÂMETROS

As configurações dos elementos que compõe a arquitetura foram feitos com auxílio de uma **RestNet**.

A partir de experimentos para determinar as melhores configurações desses componentes para **estabilidade** e **plasticidade** (muitas vezes concorrentes), as variações testadas incluem:

- Substituição a **convolução** com ***stride*** por ***average/max pooling***
- Remoção das conexões residuais e Global Average Pooling

Mais especificamente, cada camada convolucional está equipada com um *kernel* de convolução **3 × 3** e é seguida por normalização em lote (***batch normalization***) e **ReLU**.

Network Components			Performance in Task IL				Performance in Class IL			
Down.	Skip	GAP	R32-LA	R32-AIA	R18-LA	R18-AIA	R32-LA	R32-AIA	R18-LA	R18-AIA
Strided Conv.	✓	✓	25.80±1.80	35.06±0.88	38.12±2.85	49.68±0.91	37.92±0.64	55.97±0.85	40.34±0.95	57.79±1.09
	✓	×	55.52±1.41	60.89±1.83	62.95±3.45	65.41±1.92	22.63±3.36	27.37±11.18	38.85±1.60	50.29±3.01
	×	✓	25.29±1.48	31.74±0.90	30.33±2.28	41.75±0.89	30.74±2.11	46.99±1.48	38.45±0.66	56.86±0.82
	×	×	38.30±1.17	45.58±1.76	57.66±1.72	62.13±1.02	27.63±2.79	38.66±6.24	33.99±3.77	45.98±4.53
Max Pooling	✓	✓	25.91±2.35	35.58±0.78	39.89±1.80	53.15±0.69	**38.27**±0.88	**56.79**±0.88	**40.50**±0.34	**59.53**±1.26
	✓	×	**57.31**±1.56	**63.35**±1.65	**63.94**±3.13	**68.74**±3.27	24.91±3.58	22.23±13.19	40.00±0.67	54.11±1.79
	×	✓	24.54±0.53	32.16±0.87	30.46±2.52	42.24±1.08	30.69±1.27	47.50±1.42	37.34±0.72	56.53±0.83
	×	×	36.53±0.85	44.72±1.31	60.16±0.88	65.41±0.69	16.67±13.00	25.64±18.06	33.92±3.98	47.83±4.96
Avg Pooling	✓	✓	24.37±1.72	35.00±0.92	38.47±2.01	53.52±0.67	38.25±0.52	56.67±0.61	39.85±1.52	57.53±1.14
	✓	×	56.16±2.93	62.60±1.74	63.92±3.22	68.56±2.78	27.62±5.27	34.69±11.64	37.35±3.96	49.74±4.24
	×	✓	23.70±1.92	31.50±0.88	30.21±1.27	42.82±0.80	30.33±2.09	46.55±1.24	37.02±0.92	55.99±0.48
	×	×	35.86±2.09	44.93±2.42	60.68±0.86	65.78±1.29	23.04±11.10	33.05±14.94	34.84±1.67	47.63±1.41

Table 1: The CL performance of the networks with different configurations of down-sampling approaches (denoted as 'Down.') and whether to use skip connections (denoted as 'Skip') and GAP or not. 'R18' and 'R32' represent networks based on ResNet-18 and ResNet-32.

SOLUÇÃO PROPOSTA - ARCHCRAFT

SELEÇÃO DE HIPERPARÂMETROS - LARGURA E PROFUNDIDADE

A Seleção da largura e da profundidade foi feita testando valores definidos pelo autor obtendo um resultado direcionado, limitando-se aos testes realizados.

Os testes adotaram **ResNet-32** em todos os casos.

Initial Width	Final Width	Performance in Task IL		Performance in Class IL	
		LA	AIA	LA	AIA
16	64	59.48±2.03	65.65±0.89	35.82±0.55	54.43±0.47
	256	71.51±0.87	73.89±0.80	36.16±0.52	55.18±0.63
32	128	68.24±0.72	71.74±0.69	39.44±0.81	57.83±0.72
	256	74.18±0.94	76.18±0.46	40.02±0.74	58.02±0.87
48	192	70.96±1.70	74.94±0.80	40.52±0.92	58.93±0.77
	256	73.12±1.46	76.64±0.63	40.53±0.75	59.08±0.67
64	256	73.14±1.51	76.61±0.66	40.97±0.56	59.48±0.85

Table 2: The CL performance on different width configurations.

SOLUÇÃO PROPOSTA - ARCHCRAFT

CONFIGURAÇÃO DO SEARCH SPACE

A estratégia de **busca evolucionária** ocorre em duas fases: **inicialização** e **evolução iterativa.**

- Primeiro, uma população inicial de arquiteturas candidatas é gerada aleatoriamente dentro do espaço de busca e avaliada quanto à sua aptidão.

- Em cada iteração da fase evolucionária, novos indivíduos (arquiteturas) são criados por mutação. O processo envolve a seleção de dois indivíduos, escolhendo o de maior aptidão como pai, copiando-o e modificando um de seus códigos aleatoriamente. Se a alteração afetar o número de unidades, os códigos de ***down-sampling*** e canais são ajustados para manter a estrutura relativa. **Após a geração dos descendentes, os indivíduos mais aptos são selecionados** para formar a próxima população. No final do processo, a arquitetura com maior aptidão é escolhida como o *design* final.

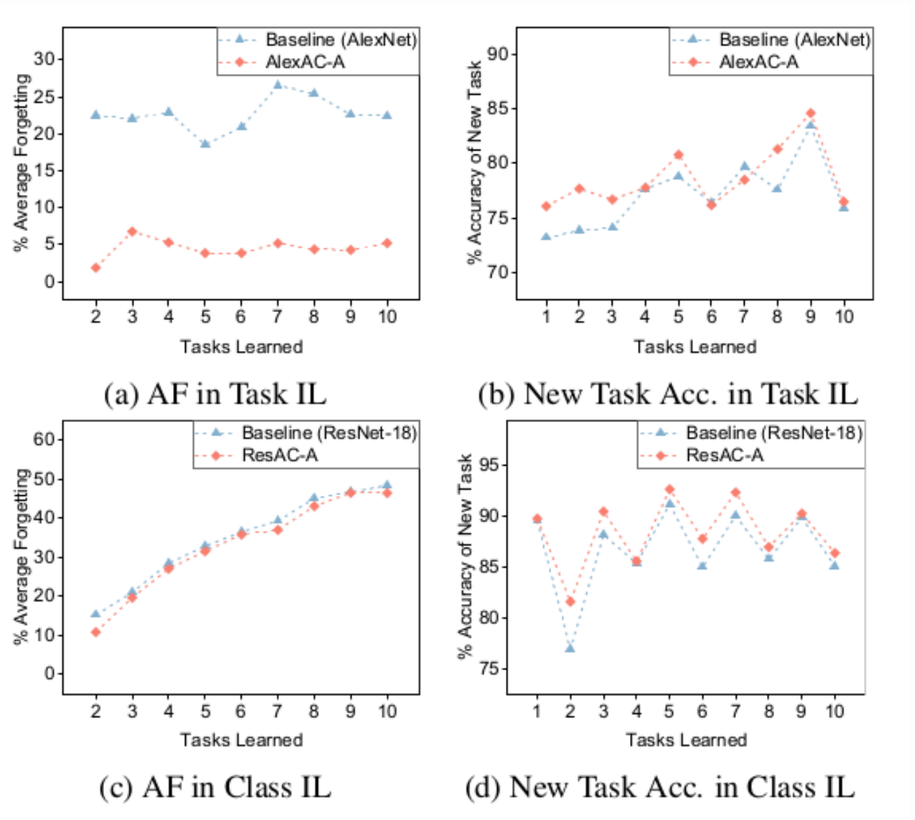
RESULTADO DO EXPERIMENTO

AVALIAÇÃO DO MODELO

A metodologia ArchCraft foi aplicada para projetar arquiteturas otimizadas para CL, utilizando uma população de **10 indivíduos e 20 gerações evolutivas**. Para avaliar sua eficácia, essas arquiteturas foram comparadas com modelos de referência.

A avaliação foi feita nos *datasets* **CIFAR-100**[1] e **ImageNet-100**[2], divididos em 20 tarefas de 5 classes cada e 10 tarefas de 10 classes cada, formando 4 benchmarks: **C100-inc5**, **C100-inc10**, **I100-inc5** e **I100-inc10**.

Como as arquiteturas foram projetadas com base no **C100-inc5**, os testes em **C100-inc10** e **ImageNet-100** verificam sua capacidade de generalização.



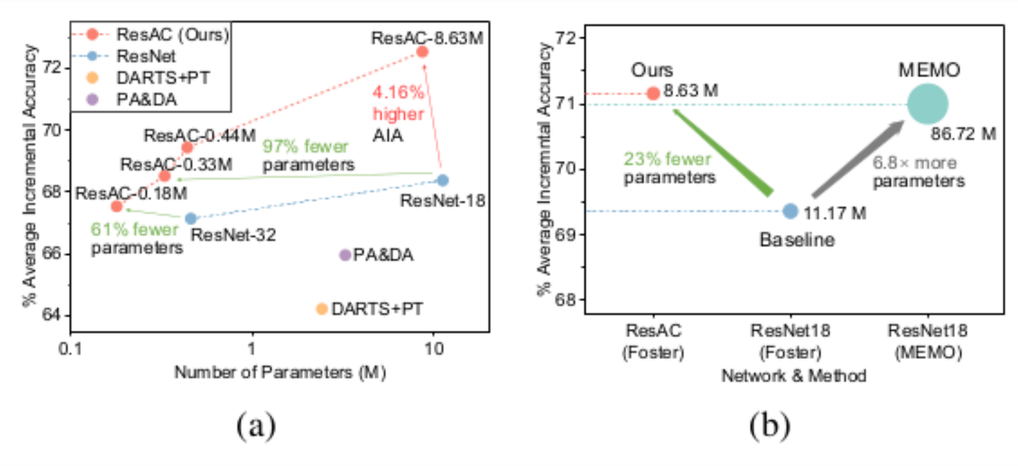
CIFAR-100, disponível em: https://www.cs.toronto.edu/~kriz/cifar.html

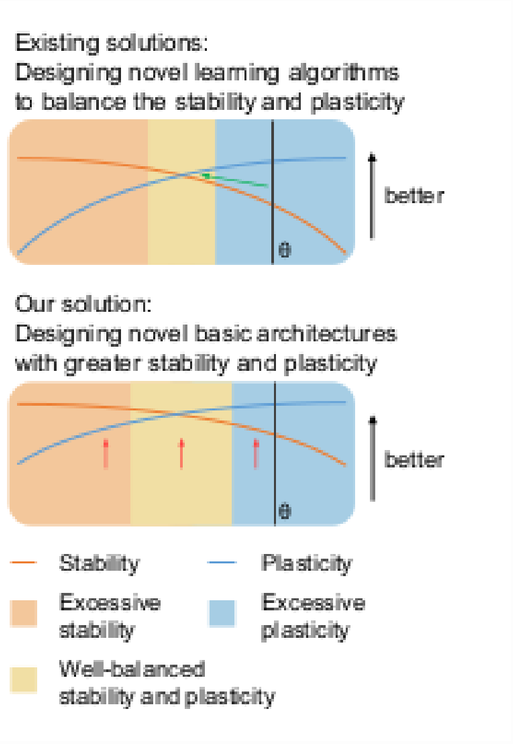
ImageNet-100, disponível em: https://www.cs.toronto.edu/~kriz/cifar.html

RESULTADO DO EXPERIMENTO

REDUÇÃO DE PARÂMETROS

1. O **ArchCraft** pode criar redes neurais com tamanhos de parâmetros controláveis, por exemplo, de **ResAC-0.18M** a **ResAC-8.63M**.
2. A série **ResAC** alcança um **desempenho superior** em aprendizado contínuo, com **significativamente menos parâmetros** do que a série ResNet tradicional. Por exemplo:
   - Com desempenho semelhante, os modelos ResAC utilizam **61% e 97% menos parâmetros** (de 0.46M para 0.18M e de 11.17M para 0.33M, respectivamente).
   - Com tamanhos de parâmetros similares, o ResAC obteve **2,3% de melhoria no desempenho do AIA**[1] (de 67,14% para 69,44%) ou mais.





1. *Average Incremental Accuracy*

RESULTADO DO EXPERIMENTO

AVALIAÇÕES PARA MÉTODOS DE CLASSIFICAÇÃO

Foram criadas duas arquiteturas baseadas nelas, **ResAC-A** e **ResAC-B**, que possuem menos parâmetros do que **ResNet-18** e **ResNet-32**.

O método proposto foi avaliado em conjunto com métodos clássicos de classificação CL, incluindo **Replay**, **iCaRL** [Rebuffi et al., 2017], **WA** [Zhao et al., 2020] e **Foster** [Wang et al., 2022].

Os hiperparâmetros de todas as abordagens seguem as configurações da biblioteca *open-source PyCIL*[1], e um tamanho de memória fixo de 2.000 exemplos foi utilizado em todos os métodos.

Method	Network	#P (M)	C100-inc5		C100-inc10		I100-inc5		I100-inc10		Max Improvement	
			LA	AIA	LA	AIA	LA	AIA	LA	AIA	LA	AIA
Replay	ResNet-32	0.46	39.10	58.17	40.02	58.21	-	-	-	-	-	-
	ResAC-B	0.44(↓ 4%)	**40.45**	**59.67**	**42.79**	**59.99**	-	-	-	-	+2.77	+1.78
	ResNet-18	11.17	40.04	58.80	43.23	60.42	36.30	57.30	41.00	59.21	-	-
	ResAC-A	8.63↓ 23%	**42.99**	**62.52**	**46.62**	**63.36**	**36.78**	**57.40**	**42.44**	**60.07**	+3.39	+3.72
iCaRL	ResNet-32	0.46	46.67	63.47	48.80	64.18	-	-	-	-	-	-
	ResAC-B	0.44↓ 4%	**47.94**	**64.17**	**50.11**	**64.42**	-	-	-	-	+1.31	+0.70
	ResNet-18	11.17	47.32	64.13	52.77	66.04	44.10	62.36	50.98	67.11	-	-
	ResAC-A	8.63↓ 23%	**52.6**	**68.71**	**55.52**	**69.62**	**45.12**	**63.98**	**52.46**	**68.42**	+5.28	+4.58
WA	ResNet-32	0.46	46.95	62.93	53.35	66.61	-	-	-	-	-	-
	ResAC-B	0.44↓ 4%	**51.31**	**66.39**	**54.89**	**67.73**	-	-	-	-	+4.36	+3.46
	ResNet-18	11.17	45.11	62.06	56.59	68.89	46.06	62.96	55.04	68.60	-	-
	ResAC-A	8.63↓ 23%	**53.23**	**69.19**	**59.79**	**71.40**	**49.94**	**67.20**	**58.86**	**71.56**	+8.12	+7.13
Foster	ResNet-32	0.46	47.78	62.36	54.36	67.14	-	-	-	-	-	-
	ResAC-B	0.44↓ 4%	**53.50**	**67.34**	**58.17**	**69.44**	-	-	-	-	+5.72	+4.98
	ResNet-18	11.17	49.03	61.97	55.98	68.38	53.26	65.20	60.58	69.36	-	-
	ResAC-A	8.63↓ 23%	**57.22**	**69.99**	**61.44**	**72.54**	**54.32**	**66.41**	**61.94**	**71.16**	+8.19	+8.02

Table 5: The CL performance of ArchCraft in *Class IL*. '#P' represents the number of parameters of the network used.

1. PyCIlL, disponiível em: https://github.com/G-U-N/PyCIL

ETAPA PRELIMINAR

1. Executar o projeto (código) sob a mesma base de dados utilizada pelo autor.
2. Reproduzir o experimento do autor, seguindo os passos descritos no artigo
3. Explicar qual é o problema que o autor encontrou nas redes neurais existentes (AlexNet e ResNet) aplicadas ao contexto de Aprendizado Contínuo.
4. Explicar como o autor simulou o aprendizado contínuo e esquematizar as etapas que que o autor seguiu.

ETAPA FINAL

1. Responder às perguntas: como o autor resolveu o problema? Qual foi o pdiferencial de seu trabalho?
2. Observar e testar se o ArchCraft obtém resultados satisfatórios em outros ***search spaces***,no contexto de aprendizado contínuo

OBRIGADO!