

## **Revisiting Neural Networks for Continual Learning: An Architectural Perspective**

**Aojun Lu<sup>1</sup>, Tao Feng<sup>2</sup>, Hangjie Yuan<sup>3</sup>, Xiaotian Song<sup>1</sup> and Yanan Sun<sup>1\*</sup>**

<sup>1</sup>Sichuan University

<sup>2</sup>Tsinghua University

<sup>3</sup>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

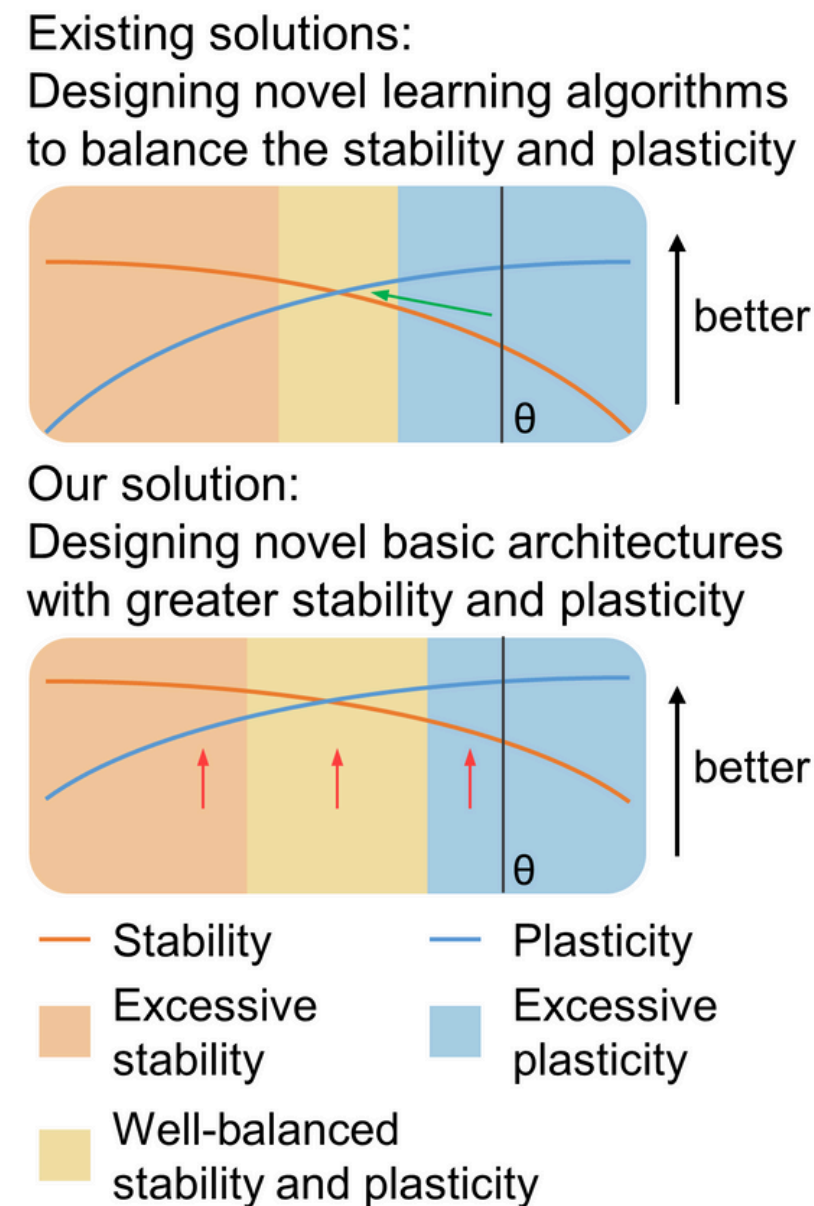
<https://arxiv.org/pdf/2404.14829>

SIN5006 - INTELIGÊNCIA COMPUTACIONAL  
ENTREGA PARCIAL

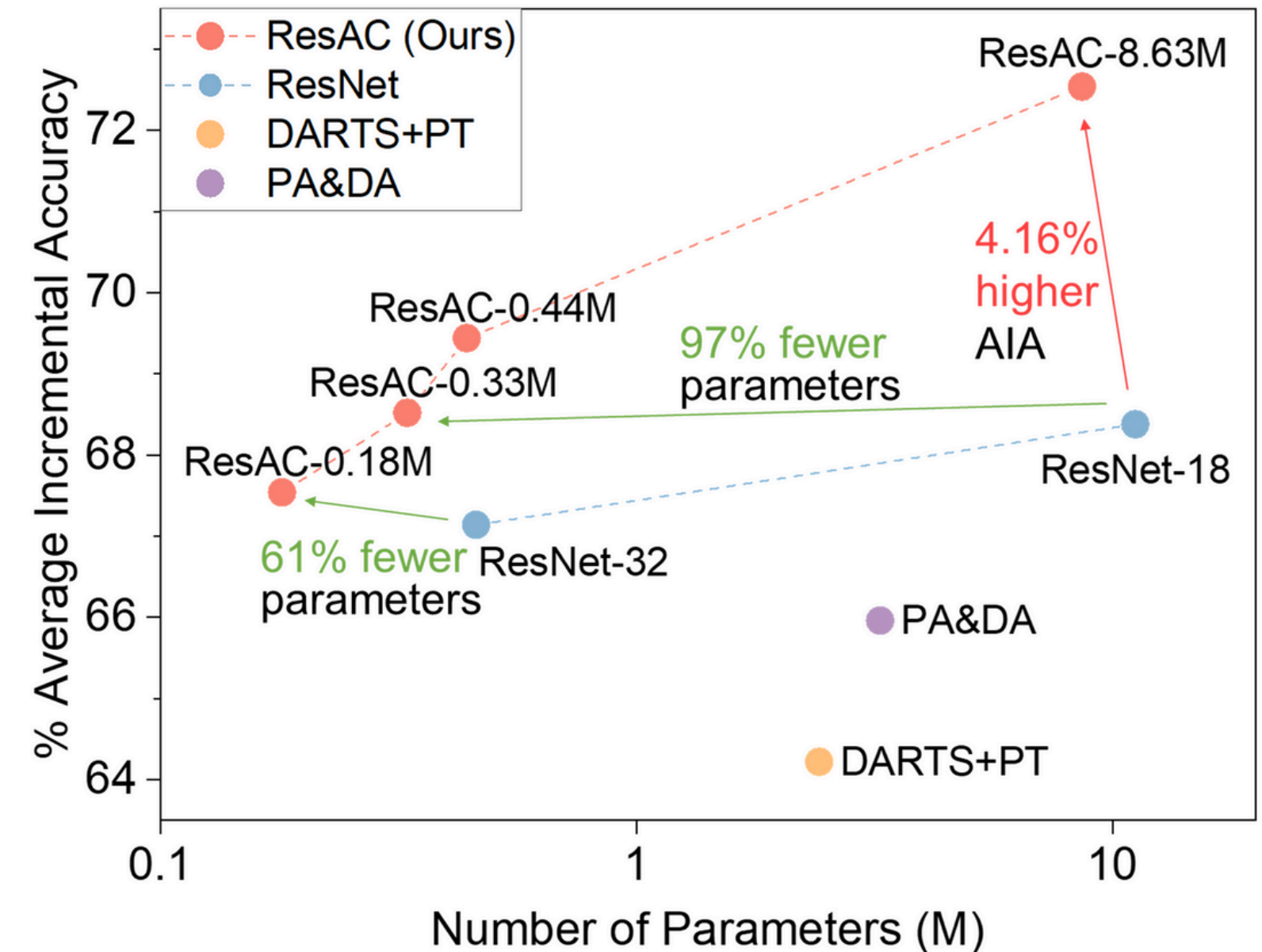
NILTON TADASHI ENTA - 12730911  
VITOR CAMARGO PINHEIRO - 12693156

# ArchCraft | Definição do problema

Apesar dos avanços em **Continuous Learning (CL)**, o autor ressalta que há **menos atenção** dada à análise do papel do **design da arquitetura de rede** na contribuição da área



Motivation of ArchCraft



CL Performance vs. Number of Parameters

# ArchCraft | Métodos (Visão Geral)

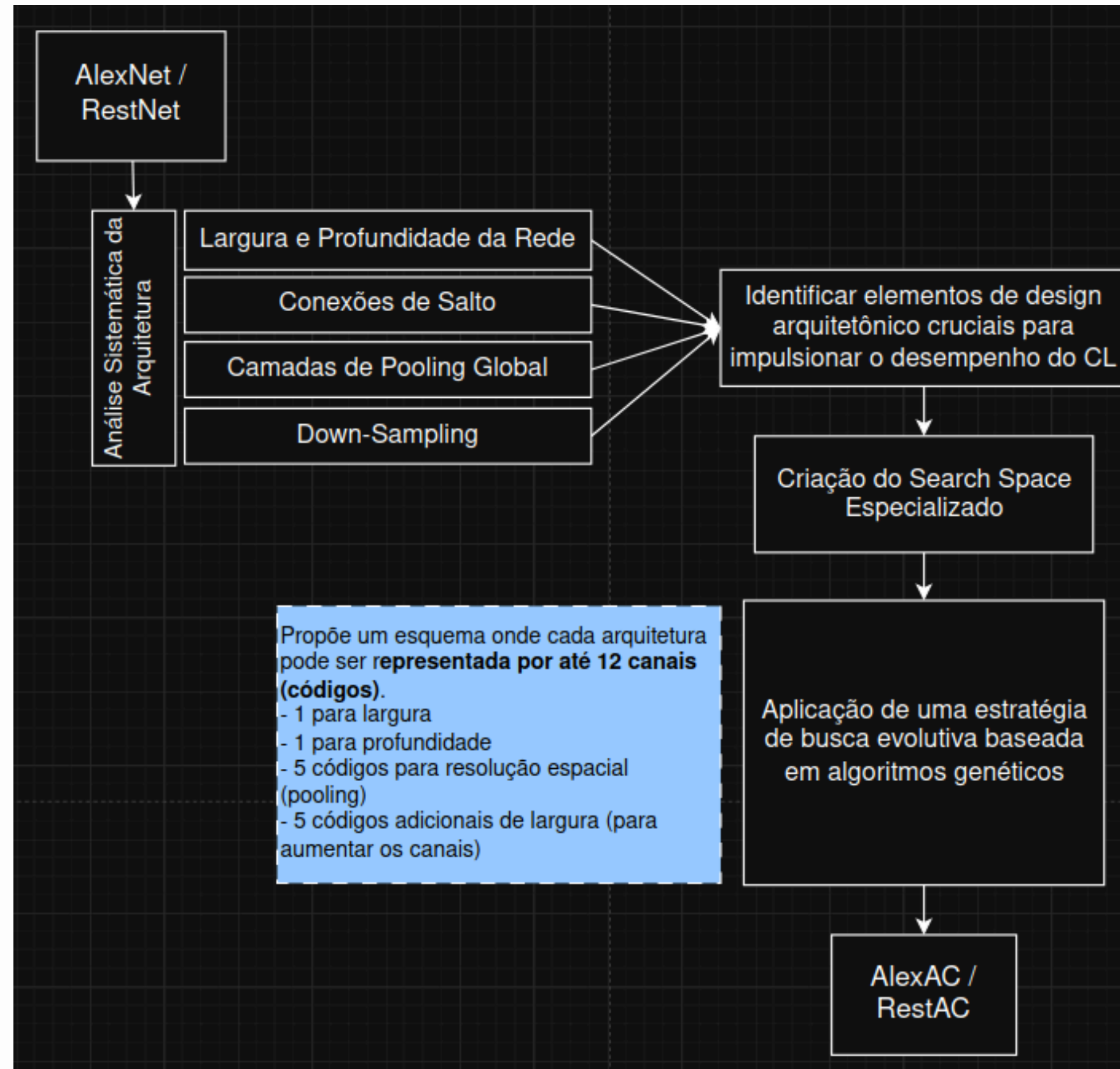
**ArchCraft** é um novo método de Neural Architecture Search (NAS).

Focado em Aprendizado Contínuo (CL) para preencher a lacuna entre o **design da arquitetura de rede e o CL**.

I. **Search Space**: Espaço de busca personalizado obtido a partir de **busca empírica** [1].

II. **Friendly Search Strategy**: Uma estratégia baseada em algoritmos genéticos para explorar o Search Space [2].

# ArchCraft | Métodos (Visão Geral)



# ArchCraft | Prometido vs Obtido

Prometido	Obtido
<ul style="list-style-type: none"><li>• Executar o projeto sob a mesma base de dados utilizada pelo autor</li></ul>	<b>EXECUTADO</b>
<ul style="list-style-type: none"><li>• Reproduzir o experimento do autor, seguindo os passos descritos no artigo</li></ul>	<b>PARCIAL:</b> Não foi reproduzido a comparação do ArchCarft com outros métodos para a tarefa de classificação

# ArchCraft | Dificuldades

- Estudo e entendimento de conceitos e técnicas relacionadas à Redes Neurais
- Entendimento da AlexNet e ResNet
- Entendimento dos benchmarks
- Entendimento do código

# ArchCraft | Desenho dos Experimentos

## Conjunto de Dados

- CIFAR100
- Incremento de 5 classes (de 100) para CL

## Medida de Avaliação

- Acurácia Incremental Média (AIA)
- Catastrophic Forgetting
- Benchmark

## Seleção de Hiperparametros

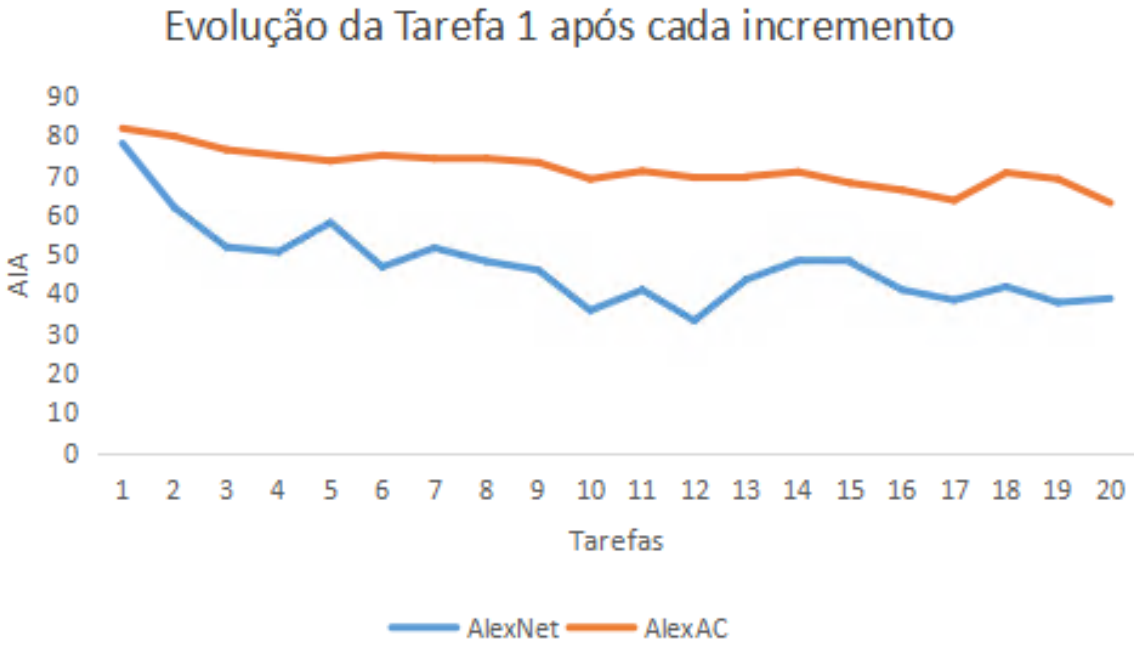
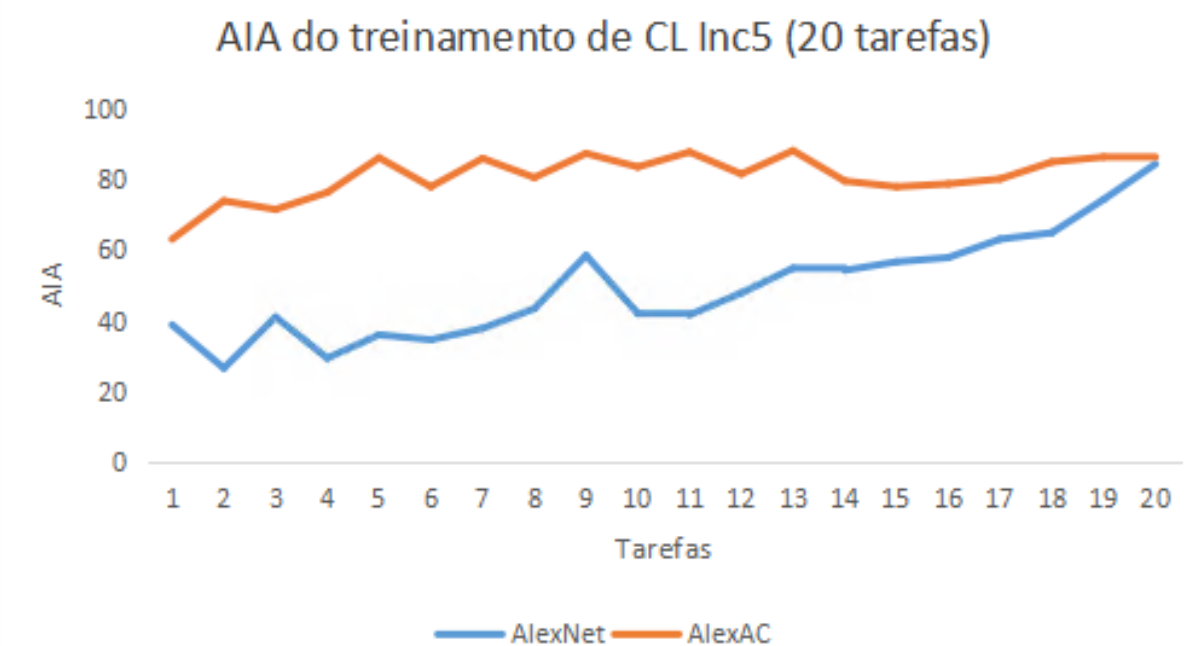
- **SGD**
  - lr: 0.01,
  - momentum: 0.9,
  - dampening: 0,
  - weight\_decay: 0.0002,
  - nesterov: False,
  - maximize: False,
  - foreach: None,
  - differentiable: False

# ArchCraft | Discussão dos Resultados

Medida	AlexNet	AlexAC	Resultado	
Acurácia Final	49,35	80,76	<b>31,41</b>	63,65 %
Cat. Forgetting	34,86	5,10	<b>-29,77</b>	-85,39 %
Params	6,71	6,28	<b>-0,43</b>	-6,43 %

Network	#P (M)	Method	C100-inc5	C100-inc10
AlexNet	6.71	Upper Bound	81.86	73.07
		SGD	53.78	56.92
		SI	70.3	62.9
		HAT	71.8	62.8
		SPG	75.9	67.7
		WSN	76.9	69.3
AlexAC-A	6.28↓ 6%	Upper Bound	83.59	75.09
		SGD	<b>82.38 (+5.48)</b>	<b>73.91 (+4.61)</b>
AlexAC-B	0.92↓ 86%	Upper Bound	83.58	74.72
		SGD	<u>79.32 (+2.42)</u>	<u>70.87 (+1.57)</u>

Table 3: The last accuracy of the AlexAC and AlexNet in *Task IL*. ‘#P’ represents the number of parameters of the network used. **Bolded** indicates best performance. Underline indicates second best.





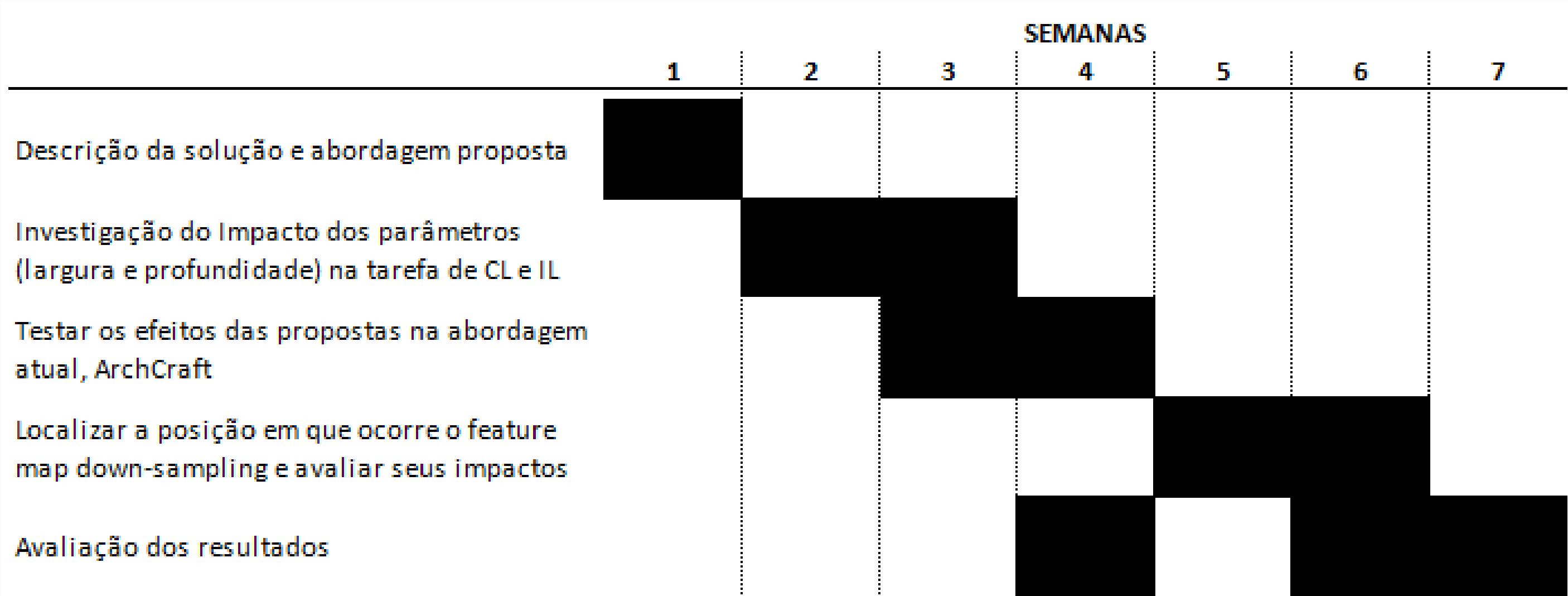
# ArchCraft | Discussão dos Resultados

```
!python test.py

2025-04-29 17:33:56,778 [trainer.py] => config: ./exps/wa.json
2025-04-29 17:33:57,034 [trainer.py] => prefix: reproduce
2025-04-29 17:33:57,034 [trainer.py] => dataset: cifar100
2025-04-29 17:33:57,034 [trainer.py] => memory_size: 2000
2025-04-29 17:33:57,035 [trainer.py] => memory_per_class: 20
2025-04-29 17:33:57,035 [trainer.py] => fixed_memory: False
2025-04-29 17:33:57,035 [trainer.py] => shuffle: True
2025-04-29 17:33:57,035 [trainer.py] => init_cls: 5
2025-04-29 17:33:57,035 [trainer.py] => increment: 5
2025-04-29 17:33:57,035 [trainer.py] => model_name: wa
2025-04-29 17:33:57,035 [trainer.py] => convnet_type: arch_craft
2025-04-29 17:33:57,035 [trainer.py] => device: [device(type='cuda', index=0)]
2025-04-29 17:33:57,035 [trainer.py] => seed: 1993
2025-04-29 17:33:57,035 [trainer.py] => depth: 10
2025-04-29 17:33:57,035 [trainer.py] => width: 144
2025-04-29 17:33:57,035 [trainer.py] => pool: [3, 7, 8, 10, 10]
2025-04-29 17:33:57,036 [trainer.py] => double: [3, 8, 10, 10, 10]
2025-04-29 17:34:03,134 [data_manager.py] => [68, 56, 78, 8, 23, 84, 90, 65, 74, 76, 40, 89, 3, 92, 55, 9, 26, 80, 43, 38, 58, 70, 77, 1, 85,
2025-04-29 17:34:10,316 [trainer.py] => All params: 8617392
2025-04-29 17:34:10,317 [trainer.py] => Trainable params: 8617392
2025-04-29 17:34:10,317 [wa.py] => Learning on 0-5
Task 0, Epoch 200/200 => Loss 0.007, Train_accy 99.96: 100% 200/200 [02:52<00:00, 1.16it/s]
2025-04-29 17:37:02,984 [wa.py] => Task 0, Epoch 200/200 => Loss 0.007, Train_accy 99.96
2025-04-29 17:37:02,985 [base.py] => Reducing exemplars...(400 per classes)
2025-04-29 17:37:02,985 [base.py] => Constructing exemplars...(400 per classes)
2025-04-29 17:37:06,563 [wa.py] => Exemplar size: 2000
2025-04-29 17:37:06,564 [trainer.py] => CNN: {'total': np.float64(96.6), '00-04': np.float64(96.6), 'old': 0, 'new': np.float64(96.6)}
2025-04-29 17:37:06,564 [trainer.py] => NME: {'total': np.float64(96.6), '00-04': np.float64(96.6), 'old': 0, 'new': np.float64(96.6)}
2025-04-29 17:37:06,564 [trainer.py] => CNN top1 curve: [np.float64(96.6)]
```

✓ 1h38m36s conclusão: 16:12

# ArchCraft | Entrega Final e Cronograma



\*Delimitar as análises para AlexNet

## ArchCraft | Referências

LU, Aojun; FENG, Tao; YUAN, Hangjie; SONG, Xiaotian; SUN, Yanan. Revisiting neural networks for continual learning: an architectural perspective. arXiv:2404.14829, 2024. Disponível em: <https://arxiv.org/abs/2404.14829>. Acesso em: 30 abr. 2025.

ANEXOS

# PSEUDOCODIGO

## DEFINE VARIÁVEIS LOCAIS

incremento <- 50

indivíduo\_id <- 0

code <- [profundidade, largura, pool[1, ..., 5], double[1, ..., 5]]

chosen\_network <- 'arch\_craft'

## Algoritmo TrainModel:

grad\_clip <- 10

epoch <- 2

learning\_rate <- 0.01

Carrega os dados CIFAR-100 divididos em tarefas incrementais

**Se** rede == 'arch\_craft': importa e instancia arquitetura Net personalizada

**Se** rede == 'alexnet': importa e instancia versão AlexNet

Mostra o número total de parâmetros do modelo

Inicializa o treinamento com otimizador SGD (Gradiente Descendente Estocástico)

Imprime a função de perda e o otimizador

Inicializa matrizes para armazenar acurácias, perdas, e medidas por tarefa

**for** tarefa **in** classes:

Treina o modelo

Salva os resultados

**for** conjunto **in** range(tarefas já processadas):

Testa a conjunto

Salva os resultados

Calcula a acurácia média após a tarefa

Calcula forgetting <-  $\text{sum}(\text{acurácia máxima} - t[x]) / \text{len}(t)$

Imprime matriz de acurácia

Calcula e imprime métrica média ap (AIA)

Calcula e imprime acurácia

**Retorna** a acurácia final

# ArchCraft | Métodos (Visão Geral)

**ArchCraft** é um novo método de Neural Architecture Search (NAS).

Focado em Aprendizado Contínuo (CL) para preencher a lacuna entre o **design da arquitetura de rede e o CL**.

**I. Search Space:** Espaço de busca personalizado obtido a partir de **busca empírica** [1].

**II. Friendly Search Strategy:** Uma estratégia baseada em algoritmos genéticos para explorar o Search Space [2].

[1] O ArchCraft ele é projetado com base em **observações empíricas sobre seus componentes arquitetônicos**.

O experimento consistiu com uma ResNet. Partindo de configurações consideradas ótimas para essas tarefas (Incremental Learning (IL)): Max Pooling, Skip connection e Global Avg. Pooling

---

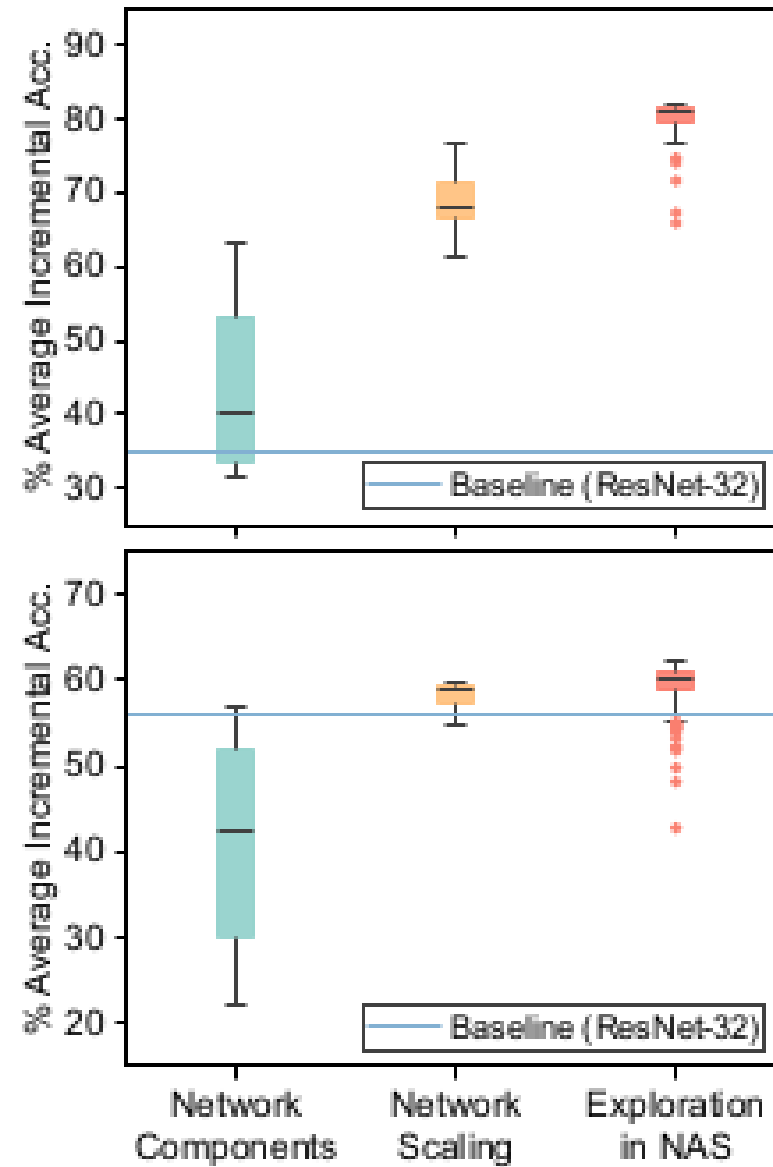
[2] **Inicialização:** Uma população de arquiteturas é gerada aleatoriamente

**Avaliação da performance:** durante as iterações evolutivas os indivíduos são avaliados por Acurácia Incremental Média (AIA).

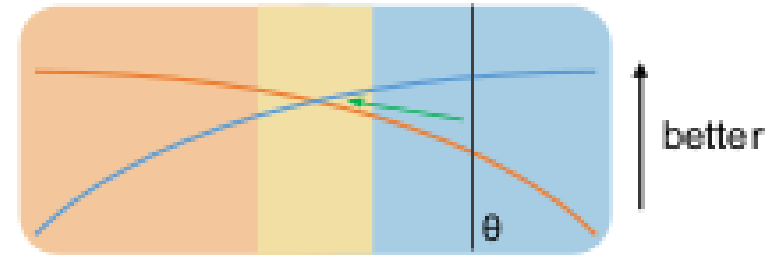
**Geração de novos indivíduos:** novas arquiteturas são geradas a partir de dois indivíduos, escolhidos aleatoriamente, chamados de filho. O filho é resultado da mutação dessas duas arquiteturas.

**Nova população:** É gerada a partir dos filhos com melhor aptidão.

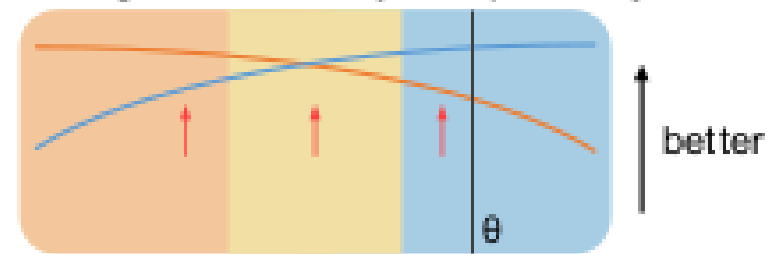
# Anexos | Resultados



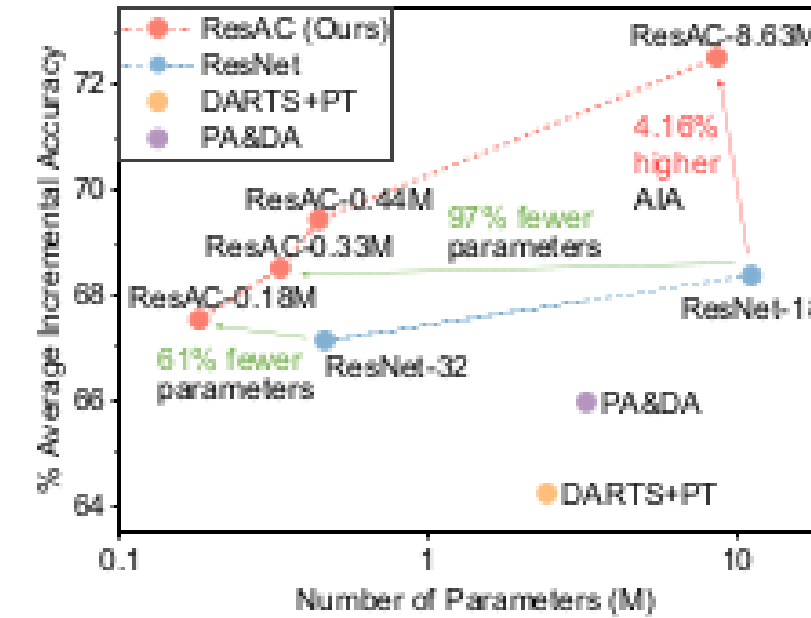
Existing solutions:  
Designing novel learning algorithms  
to balance the stability and plasticity



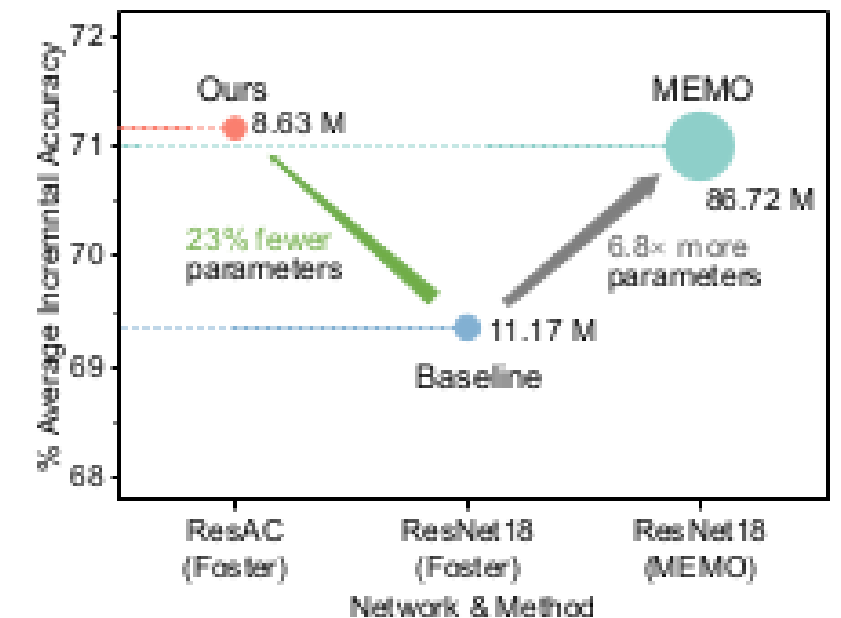
Our solution:  
Designing novel basic architectures  
with greater stability and plasticity



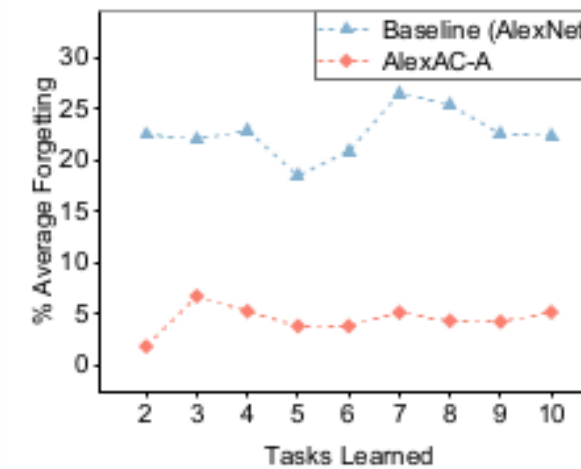
- Stability
- Excessive stability
- Well-balanced stability and plasticity
- Plasticity
- Excessive plasticity



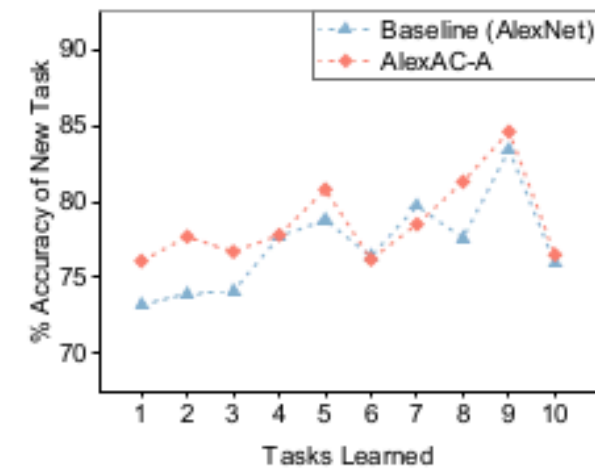
(a)



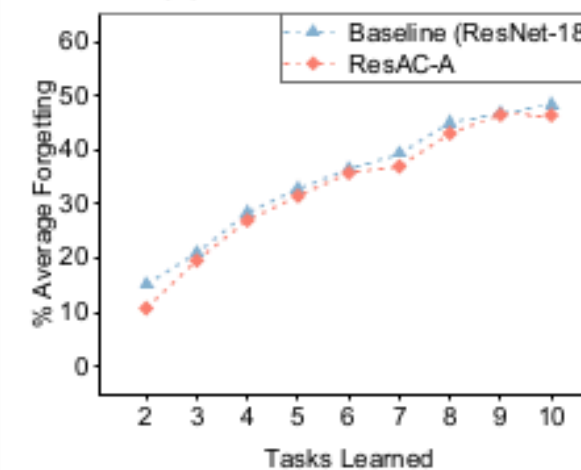
(b)



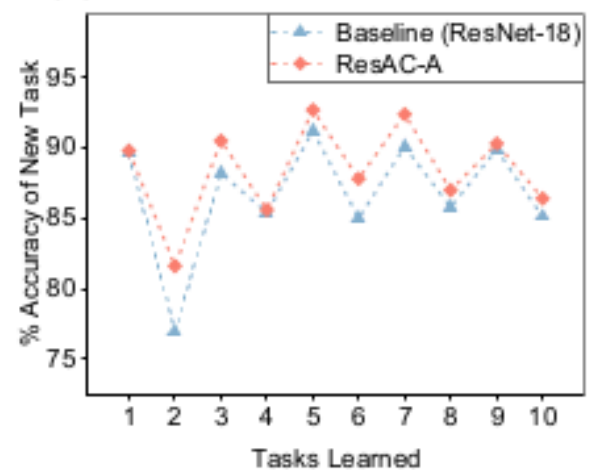
(a) AF in Task IL



(b) New Task Acc. in Task IL



(c) AF in Class IL



(d) New Task Acc. in Class IL



# Anexos | Resultados

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%)	<b>40.45</b>	<b>59.67</b>	<b>42.79</b>	<b>59.99</b>	-	-	-	-	<b>+2.77</b>	<b>+1.78</b>
	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%	<b>42.99</b>	<b>62.52</b>	<b>46.62</b>	<b>63.36</b>	<b>36.78</b>	<b>57.40</b>	<b>42.44</b>	<b>60.07</b>	<b>+3.39</b>	<b>+3.72</b>
iCaRL	ResNet-32	0.46	46.67	63.47	48.80	64.18	-	-	-	-	-	-
	ResAC-B	0.44↓ 4%	<b>47.94</b>	<b>64.17</b>	<b>50.11</b>	<b>64.42</b>	-	-	-	-	<b>+1.31</b>	<b>+0.70</b>
	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%	<b>52.6</b>	<b>68.71</b>	<b>55.52</b>	<b>69.62</b>	<b>45.12</b>	<b>63.98</b>	<b>52.46</b>	<b>68.42</b>	<b>+5.28</b>	<b>+4.58</b>
WA	ResNet-32	0.46	46.95	62.93	53.35	66.61	-	-	-	-	-	-
	ResAC-B	0.44↓ 4%	<b>51.31</b>	<b>66.39</b>	<b>54.89</b>	<b>67.73</b>	-	-	-	-	<b>+4.36</b>	<b>+3.46</b>
	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%	<b>53.23</b>	<b>69.19</b>	<b>59.79</b>	<b>71.40</b>	<b>49.94</b>	<b>67.20</b>	<b>58.86</b>	<b>71.56</b>	<b>+8.12</b>	<b>+7.13</b>
Foster	ResNet-32	0.46	47.78	62.36	54.36	67.14	-	-	-	-	-	-
	ResAC-B	0.44↓ 4%	<b>53.50</b>	<b>67.34</b>	<b>58.17</b>	<b>69.44</b>	-	-	-	-	<b>+5.72</b>	<b>+4.98</b>
	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%	<b>57.22</b>	<b>69.99</b>	<b>61.44</b>	<b>72.54</b>	<b>54.32</b>	<b>66.41</b>	<b>61.94</b>	<b>71.16</b>	<b>+8.19</b>	<b>+8.02</b>

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