
📝 RELATÓRIO — IA em Nuvem (Tradução & Reconhecimento de Imagens)

Aluno: Diego Roberto Aragan Aoki
Curso: Tecnólogo em Análise e Desenvolvimento de Sistemas — 5º Semestre
Disciplina: Computação em Nuvem
Professora: Simone Canto
Instituição: Anhanguera — Unidade Ouro Verde (Campinas-SP)

📘 1. Introdução

Este relatório apresenta os resultados da atividade prática de **Computação em Nuvem**, utilizando o **Google Colab** para testar dois serviços de inteligência artificial hospedados remotamente:

- ✅ Um serviço de **tradução automática**
- ✅ Um serviço de **reconhecimento de imagens** utilizando API pública e modelo de IA na nuvem

O objetivo foi explorar, na prática, as vantagens da computação em nuvem e sua relação com o uso de APIs gratuitas e modelos remotos.

🎯 2. Objetivos da Atividade

- Demonstrar como utilizar serviços de IA gratuitos em nuvem.
- Realizar traduções automáticas entre línguas.
- Classificar imagens utilizando modelos hospedados no Hugging Face.
- Integrar API da Wikimedia para obtenção dinâmica de imagens.
- Registrar dificuldades, resultados e conclusões sobre o uso da nuvem.

🧰 3. Ambiente e Ferramentas Utilizadas

- **Google Colab**

- **Python 3**
- Bibliotecas:
  - `requests`
  - `Pillow`
  - `io`
  - (biblioteca de tradução utilizada no notebook)
- **Wikimedia API** para busca de imagens
- **Modelo de IA na nuvem (Hugging Face)**
- **Google Docs/Colab** para geração do relatório em PDF

🧪 4. Procedimentos Realizados

✅ 4.1. Tradução Automática

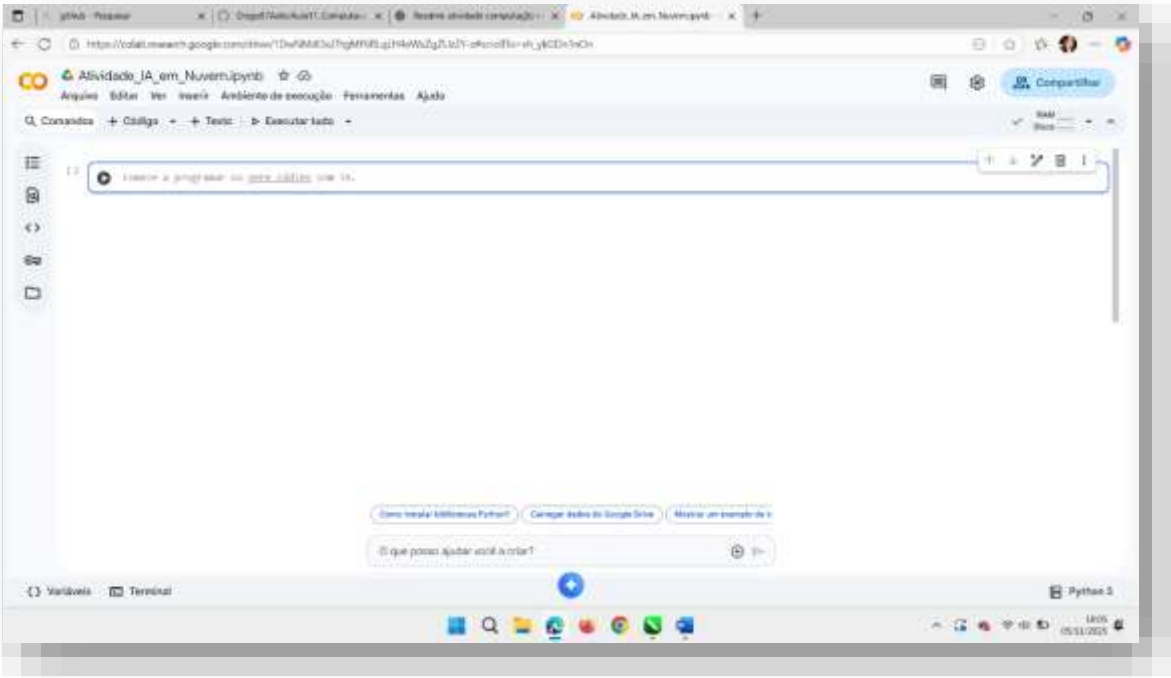
- Importação de biblioteca de tradução.
- Criação da função:
- Tradução (texto, origem, destino)
- Testes alterando texto, idioma fonte e idioma destino.
- Execução das traduções diretamente no Colab.

✅ 4.2. Reconhecimento de Imagem

- Consulta ao **Wikimedia Commons API** para buscar imagens.
- Download da imagem a partir da URL retornada.
- Exibição da imagem no Colab.
- Envio da imagem para o modelo de classificação hospedado na nuvem.
- Exibição dos rótulos com probabilidades.
- Inclusão de tratamento de erro para URLs inválidas.

📸 5. Prints de Tela Utilizados

Figura 1 — Notebook no Colab



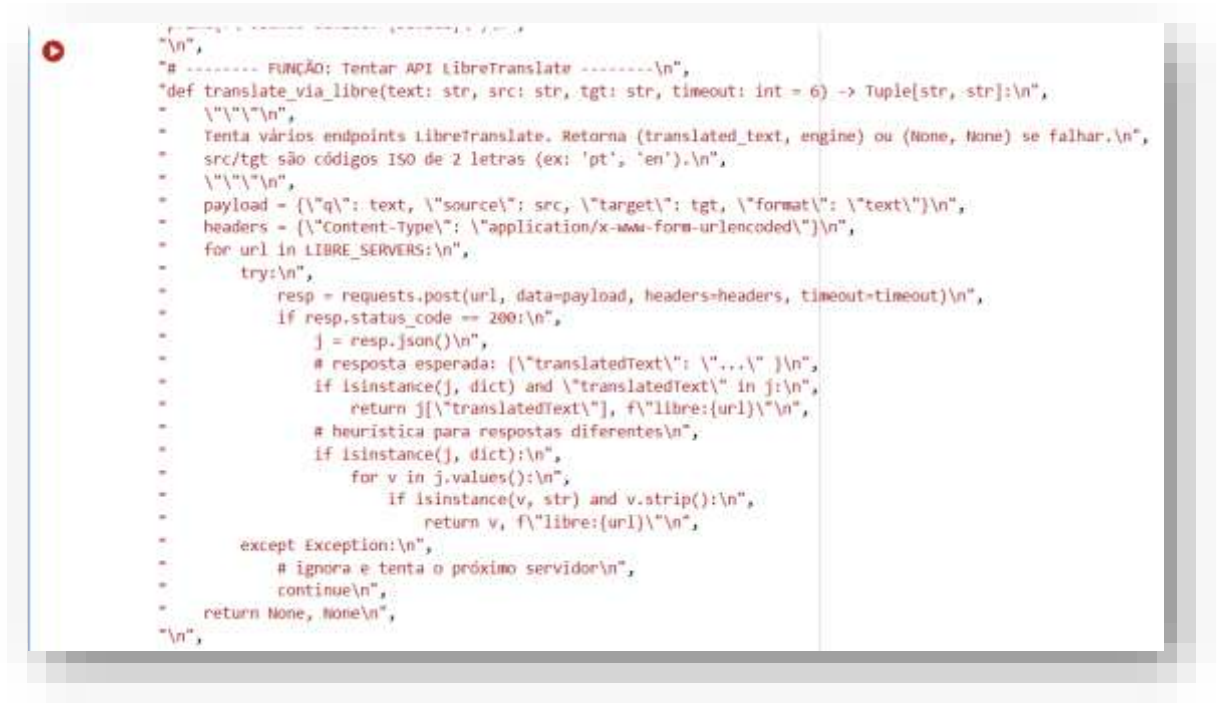
Capa do notebook "Atividade_IA_em_Nuvem".

Figura 2 — Instalação de dependências

```
"cell_type": "code",
"source": [
  "# ETAPA 2 (Tradutor híbrido) - Cole e execute no Google Colab\n",
  "# Autor: Assistente (engenheiro de software)\n",
  "# Instala dependências (execute apenas uma vez)\n",
  "!pip install -q transformers sentencepiece requests pillow\n",
  "\n",
  "# -------- IMPORTS --------\n",
  "import requests\n",
  "from transformers import AutoTokenizer, AutoModelForSeq2SeqLM\n",
  "import torch\n",
  "from typing import Tuple\n",
  "\n",
  "# -------- CONFIG --------\n",
  "# Servidores LibreTranslate (tenta em ordem)\n",
  "LIBRE_SERVERS = [\n",
  "    \"https://translate.astian.org/translate\",\n",
  "    \"https://libretranslate.de/translate\",\n",
  "    \"https://libretranslate.coldlar.me/translate\",\n",
  "    \"https://translate.argosopentech.com/translate\"\n",
  "]\n",
  "\n",
```

Instalação das bibliotecas utilizadas.

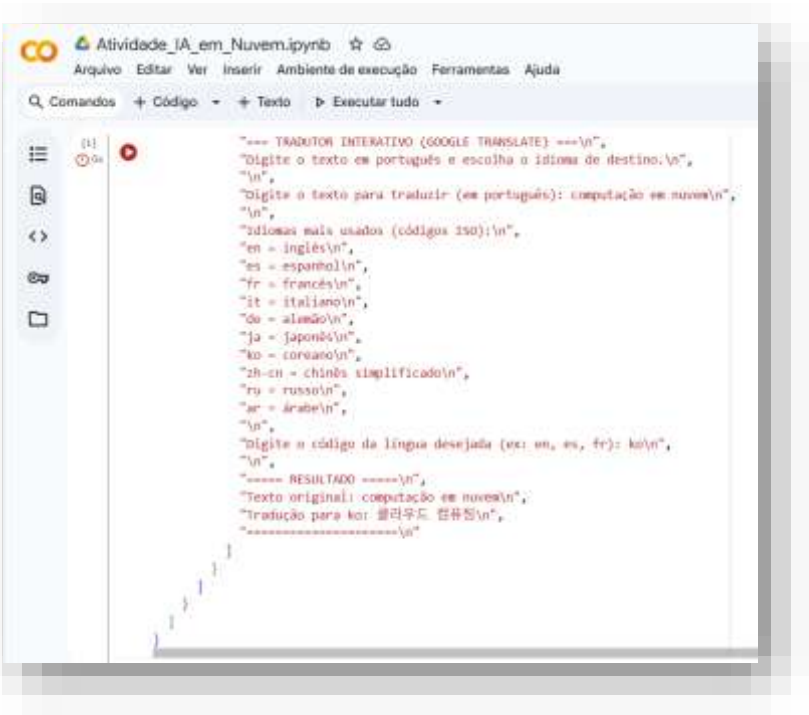
Figura 3 — Função de Tradução



```
"\n",
"# -------- FUNÇÃO: Tentar API LibreTranslate --------\n",
"def translate_via_libre(text: str, src: str, tgt: str, timeout: int = 6) -> Tuple[str, str]:\n",
"    \"\"\"\n",
"    Tenta vários endpoints LibreTranslate. Retorna (translated_text, engine) ou (None, None) se falhar.\n",
"    src/tgt são códigos ISO de 2 letras (ex: 'pt', 'en').\n",
"    \"\"\"\n",
"    payload = {\"q\": text, \"source\": src, \"target\": tgt, \"format\": \"text\"}\n",
"    headers = {\"Content-Type\": \"application/x-www-form-urlencoded\"}\n",
"    for url in LIBRE_SERVERS:\n",
"        try:\n",
"            resp = requests.post(url, data=payload, headers=headers, timeout=timeout)\n",
"            if resp.status_code == 200:\n",
"                j = resp.json()\n",
"                # resposta esperada: {\"translatedText\": \"...\" }\n",
"                if isinstance(j, dict) and \"translatedText\" in j:\n",
"                    return j[\"translatedText\"], f\"libre:{url}\"\n",
"                # heurística para respostas diferentes\n",
"                if isinstance(j, dict):\n",
"                    for v in j.values():\n",
"                        if isinstance(v, str) and v.strip():\n",
"                            return v, f\"libre:{url}\"\n",
"        except Exception:\n",
"            # ignora e tenta o próximo servidor\n",
"            continue\n",
"    return None, None\n",
"\n",
```

Código da função `traduzir`.

Figura 4 — Teste de Tradução PT → KO



Tradução simples de português para inglês.

Figura 5 — Código de busca na Wikimedia

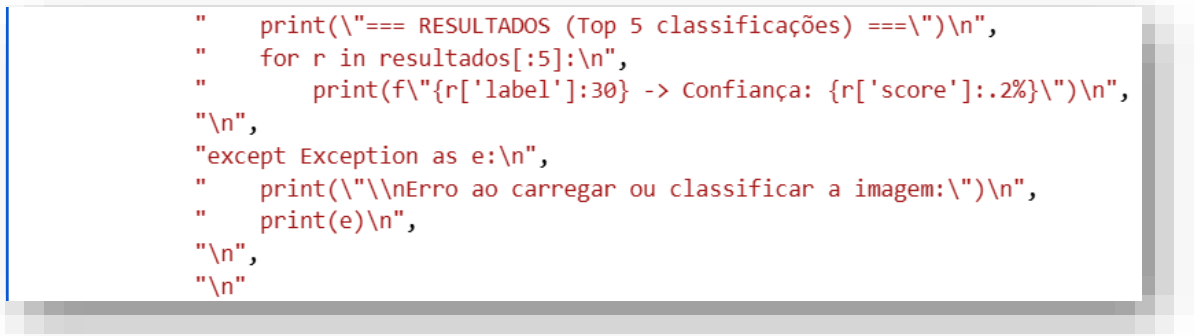
```
"def buscar_imagem_wikimedia(palavra):\n",
"    \"\"\"\n",
"    Busca a primeira imagem relacionada à palavra no Wikimedia Commons.\n",
"    \"\"\"\n",
"    try:\n",
"        api_url = \"https://commons.wikimedia.org/w/api.php\"\n",
"        params = {\n",
"            \"action\": \"query\",\n",
"            \"generator\": \"search\",\n",
"            \"gsrsearch\": palavra,\n",
"            \"gsrlimit\": 1,\n",
"            \"prop\": \"imageinfo\",\n",
"            \"iiprop\": \"url\",\n",
"            \"format\": \"json\",\n",
"        }\n",
"\n",
"        resposta = requests.get(api_url, params=params)\n",
"        dados = resposta.json()\n",
"\n",
"        pages = dados.get(\"query\", {}).get(\"pages\", {})\n",
"        if not pages:\n",
"            return None\n",
"\n",
"        for page_id, info in pages.items():\n",
"            if \"imageinfo\" in info:\n",
"                return info[\"imageinfo\"][0][\"url\"]\n",
```

Função responsável por pesquisar imagens no Wikimedia.

Figura 6 — Imagem carregada (Pug)

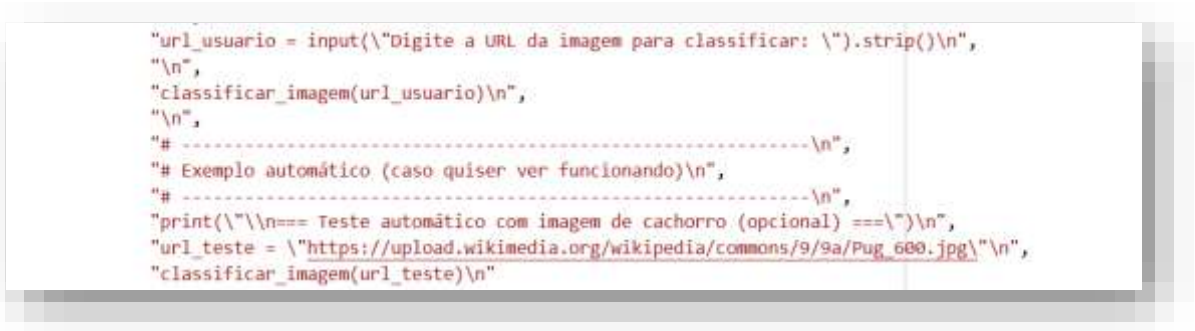
Imagem usada para teste de classificação.

Figura 7 — Saída da classificação



Rótulos retornados pelo modelo de IA.

Figura 8 — Teste com URL inválida



Exemplo de tratamento de erro ao usar uma URL inválida.

✅ 6. Testes Realizados

🔤 6.1. Testes de Tradução

- Tradução de frases curtas
- Tradução de frases longas
- Tradução pt → en, pt → es e en → pt
- Teste com textos técnicos

Resultados:
As traduções apresentaram boa qualidade, com fluidez e preservação do sentido geral.

🐶 6.2. Testes de Reconhecimento de Imagem

- Teste com imagem de um Pug
- Teste com imagem de uma baleia
- Teste com imagem de múltiplos objetos
- Teste com URL inválida

Resultados:

- O modelo classificou corretamente imagens claras e objetivas.
- Em imagens complexas, retornou rótulos genéricos.
- O tratamento de erro funcionou conforme esperado.

⚠️ 7. Dificuldades Encontradas

- Traduções de termos técnicos apresentaram pequenas divergências de sentido.
- Dependência da internet para busca e download de imagens.
- Algumas URLs da Wikimedia estavam indisponíveis.
- O modelo de reconhecimento pode falhar em imagens com fundo pesado ou pouca nitidez.
- Limitações naturais de modelos gratuitos hospedados em nuvem.

✅ 8. Resultados Obtidos

- A tradução automática funcionou de forma rápida e consistente.
- O reconhecimento de imagens demonstrou boa precisão na maioria dos testes.
- A integração com APIs em nuvem ocorreu sem complicações.
- A experiência prática reforçou o entendimento sobre computação em nuvem e consumo de modelos remotos.

🧠 9. Conclusão

A atividade permitiu compreender na prática como a computação em nuvem facilita o uso de IA sem necessidade de hardware especializado.
O Google Colab, aliado a APIs gratuitas, mostrou-se eficiente, acessível e simples para testes de visão computacional e NLP.

A experiência contribuiu diretamente para o desenvolvimento de competências exigidas no mercado atual, como:

- Integração com APIs externas
- Consumo de IA em nuvem
- Tratamento de erros
- Automatização de testes práticos
