Fase 1: Análise Exploratória e Caracterização da Fonte

Antes de escrever qualquer código de compressão, é fundamental perceber o que está dentro do ficheiro `model.safetensors`.

- **Objetivo 1.1: Engenharia Reversa da Estrutura:**
  - Analisar o formato `safetensors`. Identificar o tamanho do cabeçalho JSON (que descreve os tensores) e onde começa o *payload* binário.
  - Determinar os tipos de dados exatos (float32, float16, bfloat16, int8?). Isso é crucial porque a compressão de *floats* exige técnicas diferentes da compressão de texto.
- **Objetivo 1.2: Análise Estatística (Teoria da Informação):**
  - Calcular a **Entropia de Shannon** (H) do ficheiro "bruto" (byte a byte) para estabelecer o limite teórico inferior de compressão sem pré-processamento. (referencia: ic-notas.pdf seção 4.2)
  - Calcular a entropia de n-ésima ordem ou condicional para verificar se existem dependências óbvias entre bytes adjacentes.
  - Gerar histogramas dos valores. Se forem *floats*, analisar a distribuição dos expoentes vs. mantissas. Normalmente, os expoentes são altamente previsíveis, enquanto as mantissas são ruidosas (aleatórias).

Fase 2: *Benchmarking* de Referência (O "Baseline")

Para saber se a solução é boa, é necessário compará-la com o que já existe.

- **Objetivo 2.1: Testes com Compressores de Propósito Geral:**
  - Correr ferramentas padrão: `gzip` (DEFLATE - LZ77+Huffman), `bzip2` (BWT), `xz` (LZMA), e `zstd`.
  - Registar métricas para cada um: Tamanho final, Tempo de compressão, Tempo de descompressão, Pico de memória usada.
- **Objetivo 2.2: Testes com Compressores Especializados:**
  - Se se confirmar que são *floats*, testar ferramentas como `fpzip` ou `zfp` (em modo lossless) para ver como se comportam com dados numéricos.

Fase 3: Desenvolvimento da Estratégia de Compressão (O "Core")

Como os dados de LLMs são tensores numéricos, dicionários como LZ77 muitas vezes falham se aplicados diretamente.

- **Objetivo 3.1: Pré-processamento e Transformação:**
  - **Byte-Splitting:** Implementar uma separação de bytes. Se tiverem números de 16 bits, separar o byte alto (MSB) do byte baixo (LSB) em *streams* diferentes. O MSB costuma ter baixa entropia (compressível), o LSB alta entropia.
  - **Codificação Preditiva (Predictive Coding):** Testar preditores lineares simples ($r_n = x_n - x_{n-1}$) nos dados. A ideia é tornar a distribuição dos erros (resíduos) mais concentrada em zero (Laplaciana), reduzindo a entropia. (referencia: ic-notas.pdf seção 6)

- **Objetivo 3.2: Codificação de Entropia:**
  - Aplicar algoritmos aos dados pré-processados.
  - **Huffman:** Rápido e simples. Bom se a distribuição for enviesada. (referencia: ic-notas.pdf seção 5.1.3)
  - **Aritmética:** Mais complexo, mas dá melhor rácio. Testar se o ganho de compressão compensa o tempo extra de computação.(referencia: ic-notas.pdf seção 5.3)
  - **RLE (Run-Length Encoding):** Verificar se existem sequências longas de zeros (comum em alguns modelos esparsos).

Fase 4: Otimização e Definição de Pontos de Operação

O enunciado pede "vários pontos de operação" (rapidez vs. rácio).

- **Objetivo 4.1: Tuning de Parâmetros:**
  - Criar configurações: "Modo Rápido" (ex: Predição simples + Huffman) vs. "Modo Compacto" (ex: Modelos de Contexto mais complexos + Aritmética).
- **Objetivo 4.2: Análise de Trade-offs:**
  - Medir o impacto na memória. Se o ficheiro tem 1GB, não podem carregar tudo para a RAM se quiserem ser eficientes. Implementar leitura/escrita em *blocos* (chunks) é essencial.

Fase 5: Relatório e Apresentação

- **Objetivo 5.1: Escrita do Relatório:**
  - Descrever a estrutura descoberta no ficheiro.
  - Justificar a escolha dos algoritmos com base na análise de entropia.
  - Apresentar tabelas comparativas (Vossa Solução vs. Gzip vs. Zstd).
- **Objetivo 5.2: Preparação da Apresentação (10 min):**
  - Focar nas decisões de engenharia: "Por que escolhemos este preditor?", "Por que separámos os bytes assim?".