15. Representation Learning

Ejercicio 15.1. De sparse a dense

(a) Analice el modelo implícito detrás de los siguientes tipos de representación vectorial, compare ventajas y desventajas en términos de **dimensionalidad** y **capacidad de representación**:

1. Vectores one-hot
2. Vectores densos en $\mathbb{R}^d$

(b) Compare formalmente las siguientes dos métricas de similitud, para eso complete una tabla con: rango de valores, memoria requerida para computarla, poder de representación capturado por dicha métrica y requerimiento de aprendizaje (necesidad de entrenamiento para dicha representación):

- Producto interno entre vectores one-hot.
- Similitud de coseno entre embeddings densos.

Hint: pueden pensar que estan representando palabras en un vocabulario.

(c) **Cálculo práctico** - Calcule el aumento de memoria porcentual al pasar de un vocabulario de 50 000 palabras con embeddings de 768 d a su equivalente one-hot almacenado en `float32`.

Ejercicio 15.2. Transfer Learning

(a) Resuma el pipeline pre-training $\rightarrow$ fine-tuning y justifique su eficiencia.

(b) Para una red de clasificación binaria simple:
Vocabulario = 70 000
Embedding size = 300 d
Capa oculta = 128 d
Longitud de entrada = 3 tokens

Calcule los parámetros entrenables.

(c) Discuta las ventajas de partir de embeddings ya pre-entrenados frente a inicializarlos aleatoriamente. Explica cómo influye esto en la cantidad de datos de entrenamiento necesarios y en el tiempo de entrenamiento del modelo.

(d) Si se congelan los pesos de los embeddings y solo se actualiza el resto de la red, ¿qué beneficios y limitaciones encuentra? Vuelva a calcular el total de parámetros que permanecen entrenables bajo esta estrategia.

(e) Compare las tres estrategias de representación pre-training, transfer learning y autoencoders: en qué consiste cada uno, en qué se parecen y en qué difieren, y en qué situaciones resulta más adecuado utilizar cada enfoque.

Ejercicio 15.3. Autoencoders clásicos

(a) Diseñar un **autoencoder sobre-completo** de una sola capa en encoder y otra en el decoder:

- Escribir las ecuaciones correspondientes para cada paso del autoencoder.
- Proponer una función de pérdida con una regularización sencilla asociada a un hiperparámetro de regularización λ y justificar por qué ayuda a evitar la solución trivial.
- Analice el efecto de variar su peso hiperparámetro sobre la representación resultante.

(b) Sugiera una **métrica extrínseca** (tarea downstream) para evaluar las representaciones aprendidas y justifique su elección.

Ejercicio 15.4. Regularización con Dropout

(a) **Intuición**: Explicar por qué Dropout se entiende como un ensamblado implícito de subredes y cómo se relaciona con el concepto de bagging.

(b) **Entrenamiento vs. inferencia**

1. ¿Qué sucede con las activaciones cuando $p = 0.5$ durante el entrenamiento?
2. ¿Por qué se re-escala en entrenamiento la salida de la capa? ¿Qué sucede durante inferencia?

(c) Considere la siguiente red de ejemplo:

`Embedding` $\rightarrow$ `Dense(128)` $\rightarrow$ `Softmax(3)`

Identifique qué capa(s) se benefician más de Dropout y justifique.

(d) **Trade-off p alto/bajo** - Describa dos escenarios donde Dropout empeore el rendimiento y proponga regularizaciones alternativas.

Ejercicio 15.5. Embeddings de palabras (Skip-Gram) En este ejercicio trabajaremos con el modelo Skip-Gram visto en clase para aprender embeddings de palabras.

(a) Escriba la estructura mínima en pseudocódigo de una red para entrenar Skip-Gram.

(b) Explique el pre-proceso necesario al entrenar con **Wikipedia**.

(c) ¿Cuántos anotadores humanos harían falta para "etiquetar" Wikipedia y entrenar Skip-Gram? Justifica tu respuesta.

(d) Muestre un ejemplo concreto antes de ser presentado a la red (input y label).

(e) Compare con un ejemplo equivalente tomado de otra red como **Reddit**.

(f) Formule la pérdida con negative sampling e identifique todos sus hiperparámetros.

(g) Explica por qué usamos negative sampling en lugar de la softmax completa.

Ejercicio 15.6. Autoencoder lineal

(a) **Arquitectura**
Diseñe un autoencoder lineal para entradas $x \in \mathbb{R}^d$:

$$h = W_e\, x, \qquad \hat{x} = W_d\, h, \qquad h \in \mathbb{R}^k \ (k < d).$$

1. Elija la dimensión k y justifique su valor en términos de tasa de compresión y reconstrucción aceptable.
2. Comente el efecto de atar los pesos ($W_d = W_e^\top$) frente a entrenarlos de forma independiente.

(b) **Función de pérdida**

1. Escriba la función de perdida mean squared error (MSE) y explique por qué es la opción estándar cuando se busca reconstruir datos continuos.
2. Mencione algunas situaciones en las que una loss como cross-entropy podría ser más apropiada.

(c) **Compresión vs. reconstrucción**
Para un conjunto de datos cualquiera, proponga un experimento conceptual (sin código) para:

1. Medir cómo varía la MSE cuando k toma los valores $\{d,\ \frac{3d}{4},\ \frac{d}{2},\ \frac{d}{4}\}$.
2. Determinar un umbral de k a partir del cual la degradación de la reconstrucción es "significativa". Explique su criterio.

(d) **Regularización**
Proponga dos técnicas de regularizacion para mejorar la estabilidad del modelo y evitar sobre-ajuste. Discuta cómo afectan la calidad de la representación h.

(e) **Evaluación extrínseca**
Elija una tarea downstream sencilla (p. ej. regresión o clasificación logística usando h) y describa:

1. ¿Como se determina el upper bound del rendimiento para cada tarea?
2. Cómo comparar el rendimiento de las representaciones del autoencoder para distintos k.
3. Qué conclusiones se pueden extraer sobre la utilidad práctica de la compresión.

Ejercicio 15.7. Autoencoder multimodal para detección de anomalías
Suponga que dispone de una tabla con N filas con diferentes columnas que tienen distintos tipos de datos como se muestra en la tabla 4.
El objetivo es entrenar un autoencoder que reconstruya cada fila y cuya reconstruction error se utilice como score de anomalía.

Tabla 4: Tipos de datos multimodales

Tipo	**Ejemplos**
Numérica	`precio`, `temperatura`, `edad`
Categórica	`país`, `forma_de_pago`
Texto corto	`comentario_cliente` (≤ 100 tokens)
Imagen	`foto_producto` (RGB, 224×224)
Binaria	`fraude_reportado`

(a) **Pre-procesamiento y codificación**
Complete la tabla indicando cómo convertir cada tipo de dato en un vector de entrada x_i y qué decodificación inversa usar para $\hat{x}_i$.

Tipo	Encoder (dim.)	Decoder / salida
Numérica		
Categórica		
Texto		
Imagen		
Binaria		

(b) **Arquitectura propuesta**
Describe a grandes rasgos una red que:

- Para cada tipo de dato usa un encoder específico que produce embeddings homogéneos de tamaño d.
- Una capa que produzca un vector latente $h \in \mathbb{R}^k$.
- Decodificadores simétricos para reconstruir cada x_i.

Justifique la elección de k y de las dimensiones intermedias.

(c) **Función de pérdida compuesta**

1. Indique una loss apropiada para cada tipo de dato.
2. Escriba la pérdida total como suma ponderada $\mathcal{L} = \sum_i \lambda_i \mathcal{L}_i(x_i, \hat{x}_i)$ y explique un criterio para fijar (o aprender) los pesos λ_i.

(d) **Plan de entrenamiento**
Detalle:

- Estrategia de batching (mezclar o no filas con imágenes faltantes, padding o unk de texto, etc.).
- Hiperparámetros clave (LR, batch, optimizador, número de épocas) y técnicas de regularización (Dropout, data augmentation en imágenes, word dropout en texto).
- Procedimiento para evitar colapso trivial (p. ej. early stopping, peso mínimo a la pérdida de cada rama).

(e) **Inferencia y umbral de anomalía**

1. Proponga una métrica escalar de error por fila (p. ej. suma o media ponderada de $\mathcal{L}_i$).
2. Describa dos métodos para fijar el umbral: (i) percentil q del error en el conjunto de entrenamiento; (ii) validación con etiquetas de anomalía conocidas.
3. Explique cómo interpretar los errores parciales por tipo para diagnosticar la fuente de la anomalía.

(f) **Evaluacion**
Describa un proceso de evaluacion que valide la red entrenada.

Ejercicio 15.8. Representaciones en GANs

(a) Describa el juego **generador–discriminador** y el concepto de equilibrio de Nash.

(b) Explique **mode collapse** y proponga dos técnicas para mitigarlo.