RESUMEN TEORÍA APRENDIZAJE AUTOMÁTICO:

ÍNDICE DE CONTENIDOS:

[] CAPÍTULO 5: COMPRESIÓN DE DATOS MEDIANTE REDUCCIÓN DE DIMENSIONALIDAD

} 5.1 INTRODUCCIÓN

Métodos de reducción de dimensionalidad: Técnicas de selección de características como feature extraction (alternativa a feature selection que implica transformar el conjunto de datos en un nuevo subespacio de características con menor dimensionalidad).

Importancia: Al transformar datos de mayor dimensión en un espacio de menor dimensión, se reduce la complejidad del conjunto de datos, lo que puede resultar en tiempos de entrenamiento bajos y un uso menor de recursos informáticos. Al descartar datos irrelevantes o ruidosos se ayuda a crear datos y modelos más sólidos. Al reducir el número de dimensiones, se pueden visualizar datos de 2 a 3 dimensiones, lo que facilita la identificación de patrones y clusters.

Feature selection: Se selecciona un subconjunto de características originales del conjunto de datos, cuyo objetivo consiste en identificar y mantener las características más relevantes que contribuyan al poder predictivo del modelo.

Feature extraction: Se transforma y se proyecta el conjunto de datos original en un nuevo espacio de características. Esto implica crear nuevas características combinando las existentes, obteniendo como resultado una dimensionalidad reducida que puede capturar información esencial del conjunto de datos.

} 5.2 FEATURE EXTRACTION

Feature extraction: Enfoque para la compresión de datos con el objetivo de mantener la mayor parte de la información relevante.

Usos: Mejorar el espacio de almacenamiento, la eficiencia computacional del aprendizaje y el rendimiento predictivo descartando información irrelevante y/o ruidosa, permitir una visualización de los datos tanto en 2 como en 3 dimensiones y reducir la llamada 'curse of dimensionality'.

Problemas: En dimensiones altas, los datos tienden a ser más escasos. Esto significa que los puntos de datos están más separados, donde la mayor parte del espacio de alta dimensión está vacío, lo que dificulta la identificación de patrones y hace que el desarrollo de clusterings y de tareas de clasificación sea más desafiante. Por tanto, en dimensiones altas se requieren más datos para llenar el espacio vacío y obtener resultados significativos. Si no se tienen más datos, los algoritmos son propensos a sobre ajustarse. En un intento por capturar toda la variabilidad del conjunto de datos, el modelo puede volverse complejo y ajustarse a datos irrelevantes y detalles que no se generalizan bien en datos nuevos.

} 5.3 PCA (PRINCIPAL COMPONENT ANALYSIS)

PCA (Principal Component Analysis): Técnica de transformación de análisis lineal no supervisado ampliamente utilizada en diferentes campos para la extracción de características y la reducción dimensional. Tiene como objetivo encontrar las direcciones de máxima varianza en datos de alta dimensión, además de proyectar los datos en un nuevo subespacio con las mismas o menos dimensiones que el original. Los ejes ortogonales del nuevo subespacio se pueden interpretar como las direcciones de máxima varianza dada la restricción de que los nuevos ejes de características son ortogonales entre sí, donde x1 y x2 son los ejes característicos originales, y PC 1 y PC 2 son los componentes principales.
Matriz de transformación W de dimensiones d × k: Permite mapear un vector de dimensión d de las características del ejemplo de entrenamiento, donde x es un nuevo subespacio de características de dimensión k, el cual tiene menos dimensiones que el espacio de características de dimensión d original.
Estandarización: Centrar los datos (restar la media de cada característica del conjunto de datos para que cada característica tenga una media de 0) y escalar los datos (dividir cada característica por su desviación estándar para que cada característica tenga una desviación estándar de 1).
z-score normalization: Al estandarizar los datos, se garantiza que cada característica contribuye por igual al cálculo de las componentes principales, permitiendo a PCA identificar las direcciones de máxima varianza sin verse sesgadas por la escala de las características.
Pasos: Paso 1 (estandarizar el conjunto de datos d-dimensional), paso 2 (construir la matriz de covarianza), paso 3 (descomponer la matriz de covarianza en vectores y valores propios), paso 4 (ordenar los valores propios en orden decreciente para clasificar los correspondientes vectores propios), paso 5 (seleccionar k vectores propios, que corresponden a los k valores propios más grandes, donde k es la dimensionalidad del nuevo subespacio de características), paso 6 (construir una matriz de proyección W a partir de los k vectores propios superiores), paso 7 (transformar el conjunto de datos de entrada d-dimensional usando la matriz de proyección W para obtener el nuevo subespacio de características k-dimensional).
Matriz de covarianza: Matriz simétrica de dimensiones d x d, donde d es el número de características en el conjunto de datos, la cual almacena las covarianzas por pares entre las diferentes características.
Vectores propios: Representan los componentes principales y las direcciones de máxima varianza.
Valores propios: Definen la magnitud de los vectores propios.
Nuevas características: Representan combinaciones lineales de las características originales con los componentes principales, donde cada Wij son las contribuciones de las funciones originales a la nueva función. Sin embargo, para medir las contribuciones de las características originales a la nueva característica, se utilizan versiones escaladas de W, donde cada vector propio se multiplica por la raíz cuadrada de su valor propio, cuyo resultado se denomina como 'carga'.
Importancia del escalado: Los valores resultantes pueden interpretarse como la correlación entre las características originales y las nuevas características.

} 5.4 LDA (LINEAR DISCRIMINANT ANALYSIS)

LDA (Linear Discriminant Analysis): Técnica de transformación lineal que toma información y tiene en cuenta las etiquetas de cada clase, y que se puede utilizar para reducir el número de dimensiones (LDA de Fisher).
Objetivos: Busca encontrar una proyección lineal de los datos que maximice la separabilidad entre diferentes clases, en otras palabras, intenta identificar una o más direcciones en el espacio de características (discriminadores lineales) que maximicen la distancia entre las medias de cada clase.
Número máximo de discriminantes lineales: Determinado por el número de clases y de características en su conjunto de datos. Específicamente, es el menor de C - 1 (número de clases en el conjunto de datos) y d (número de características originales). Por tanto, su máximo es min(C - 1, d).
Pasos: Paso 1 (estandarizar el conjunto de datos d-dimensional, donde d es el número de características), paso 2 (calcular el vector medio d-dimensional para cada clase), paso 3 (construir la matriz de dispersión entre clases Sb y la matriz de dispersión dentro de clases Sw), paso 4 (calcular los vectores y valores propios correspondientes de la matriz Sw ^ -1 * Sb), paso 5 (ordenar los valores propios en orden decreciente para clasificar los vectores propios correspondientes), paso 6 (elegir los k vectores propios que correspondan a los k valores propios más grandes para construir una matriz de transformación W de dimensiones d x k, donde los vectores propios son las columnas de esta matriz) y paso 7 (proyectar los ejemplos en el nuevo subespacio de características usando la matriz de transformación W).

} 5.5 T-SNE (T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING)

t-SNE (t-distributed Stochastic Neighbor Embedding): Utilizada para visualizar conjuntos de datos de alta dimensión en 2 o 3 dimensiones, aprende a incrustar puntos de datos en una dimensión espacial inferior tal que las distancias por pares en el espacio original se conservan, es una técnica destinada a fines de visualización, ya que requiere todo el conjunto de datos para la proyección. Dado que los puntos se proyectan directamente, a diferencia de PCA que no involucra una matriz de proyección, no se puede aplicar t-SNE a nuevos puntos de datos.

Reducción dimensional no lineal: Muchos algoritmos de aprendizaje automático hacen suposiciones sobre la separabilidad lineal de los datos de entrada. Sin embargo, si se están tratando problemas no lineales encontrados con bastante frecuencia en aplicaciones del mundo real, existen técnicas de transformación lineal para la reducción de dimensionalidad como PCA, aunque LDA no puede ser la mejor opción.

[] CAPÍTULO 6: APRENDIZAJE DE PRÁCTICAS PARA LA EVALUACIÓN DE MODELOS Y EL AJUSTE DE HIPERPARÁMETROS

} 6.1 HIPERPARÁMETROS

Hiperparámetros: Valores que controlan el comportamiento del modelo de Machine Learning, pero no se aprenden directamente de los datos. Es decir, se establecen antes del entrenamiento y afectan a cómo el modelo aprende (tasa de aprendizaje en redes neuronales, número de árboles en un Random Forest, profundidad máxima en un árbol de decisión y parámetro de regularización en SVM).
Parámetros: Valores aprendidos por el modelo durante el entrenamiento (pesos y sesgos en red neuronal).
Hiperparámetros: Valores externos que no se aprenden, pero afectan el proceso de aprendizaje (número de capas en una red neuronal).
Sobreajuste (overfitting): Si el modelo es demasiado complejo (demasiados árboles o capas), puede ajustarse demasiado a los datos de entrenamiento y no generalizar bien.
Subajuste (underfitting): Si el modelo es demasiado simple (pocos árboles o baja profundidad), no capturará los patrones importantes de los datos. Una buena configuración de hiperparámetros puede mejorar la precisión del modelo, reducir el tiempo de entrenamiento y lograr un balance entre sobreajuste y subajuste.
Búsqueda de hiperparámetros: Proceso de encontrar la mejor combinación de hiperparámetros para optimizar el rendimiento del modelo con el objetivo de maximizar la métrica de evaluación (accuracy, F1-score, AUC) en un conjunto de validación o mediante validación cruzada.
Problema: Elegir los hiperparámetros de manera manual (prueba y error) es ineficiente y poco sistemático.
Solución: Usar métodos como Grid Search, Random Search o técnicas avanzadas para automatizar y optimizar este proceso.
División de los datos: Evalúa el rendimiento del modelo de manera justa y evitar el sobreajuste con el objetivo de garantizar que el modelo generalice bien en datos nuevos no vistos durante el entrenamiento.

} 6.2 TIPOS DE PARTICIONES

Conjunto de entrenamiento (training): Entrena el modelo (pesos en una red neuronal) y representa la mayor parte de los datos (60-80% del total).
Conjunto de validación (validation): Evalúa el modelo durante el entrenamiento y ajusta los hiperparámetros, ayuda a seleccionar el mejor modelo o configuración sin usar el conjunto de prueba y representa típicamente el 10-20% de los datos (evita sobreajustes del conjunto de entrenamiento).
Conjunto de prueba (test): Evalúa el rendimiento del modelo final en datos completamente nuevos y representa el 10-20% de los datos.

} 6.3 BÚSQUEDA DE HIPERPARÁMETROS CON VALIDACIÓN

Conjunto de validación: Paso 1 (se dividen los datos en entrenamiento, validación y prueba), paso 2 (se entrena el modelo con el conjunto de entrenamiento con una combinación de hiperparámetros), paso 3 (se evalúa el modelo con el conjunto de validación para cada combinación de hiperparámetros), paso 4 (se selecciona la combinación que maximiza la métrica de evaluación como accuracy o F1-score), paso 5 (una vez seleccionados los mejores hiperparámetros, se entrena el modelo final con los datos de entrenamiento y validación combinados) y paso 6 (se evalúa el modelo final en el conjunto de prueba).

} 6.4 VALIDACIÓN CRUZADA

K-fold cross-validation: Técnica que permite evaluar el modelo de manera más robusta cuando los datos son limitados. En lugar de usar un conjunto de validación fijo, se divide el conjunto de entrenamiento en múltiples particiones (o "folds") y se entrena y valida el modelo varias veces.
Pasos: Paso 1 (se dividen los datos en entrenamiento y prueba), paso 2 (se entrena el modelo con el conjunto de entrenamiento con una combinación de hiperparámetros, es decir, para cada combinación de hiperparámetros, se entrena el modelo k veces, utilizando k-1 folds para entrenamiento y 1 fold para validación; se calcula la métrica de evaluación como accuracy o F1-score en cada fold; se promedian las métricas obtenidas en los k-folds para obtener una evaluación final de esa combinación de hiperparámetros), paso 3 (se selecciona la combinación de hiperparámetros que maximiza la métrica promedio), paso 4 (una vez seleccionados los mejores hiperparámetros, se entrena el modelo final con los datos de entrenamiento completo) y paso 5 (se evalúa el modelo final en el conjunto de prueba).

} 6.5 MÉTODOS DE BÚSQUEDA DE HIPERPARÁMETROS

Búsqueda de hiperparámetros: Consiste en encontrar la mejor combinación de configuraciones que optimicen el rendimiento de un modelo de Machine Learning.
Búsqueda manual: Consiste en ajustar los hiperparámetros manualmente, probando diferentes combinaciones basadas en la intuición, experiencia o conocimiento del problema. Por ejemplo, probar diferentes tasas de aprendizaje o profundidades de árbol y observar cómo afecta al rendimiento del modelo.
Ventajas / Desventajas: Ineficiente, poco sistemático y difícil de aplicar cuando hay muchos hiperparámetros o combinaciones posibles y depende de la experiencia del usuario, lo que puede llevar a resultados subóptimos.
Grid Search: Busca en un espacio definido de combinaciones de hiperparámetros en el que se define un rango o conjunto de valores posibles para cada hiperparámetro, y el método prueba todas las combinaciones posibles. Por ejemplo, si se tienen 2 hiperparámetros con 3 valores cada uno, Grid Search probará las 9 combinaciones posibles.
Ventajas: Garantiza encontrar la mejor combinación dentro del espacio definido, fácil de implementar y paralelizar y es ideal para espacios de búsqueda pequeños y bien definidos.
Desventajas: Computacionalmente costoso, especialmente cuando hay muchos hiperparámetros o valores posibles (combinación explosiva) y no es eficiente si algunos hiperparámetros tienen menor impacto en el rendimiento.
Random Search: Selecciona combinaciones aleatorias de hiperparámetros dentro de un rango definido. En lugar de probar todas las combinaciones posibles (Grid Search), selecciona un número fijo de combinaciones aleatorias.
Ventajas: Más eficiente que Grid Search, especialmente cuando algunos hiperparámetros tienen menor impacto en el rendimiento, puede explorar un espacio de búsqueda más amplio en menos tiempo y reduce el costo computacional al limitar el número de combinaciones probadas.
Desventajas: No garantiza encontrar la mejor combinación de hiperparámetros y puede requerir muchas iteraciones para obtener buenos resultados si el espacio de búsqueda es muy grande.
Búsqueda bayesiana: Utiliza modelos probabilísticos para explorar el espacio de búsqueda de manera más inteligente. En lugar de probar combinaciones al azar o exhaustivamente, la búsqueda bayesiana utiliza los resultados de combinaciones anteriores para predecir qué combinaciones podrían ser más prometedoras (algoritmos como Tree-structured Parzen Estimators o Gaussian Processes).
Ventajas / Desventajas: Más eficiente que Grid Search y Random Search, ya que se enfoca en las regiones del espacio de búsqueda con mayor probabilidad de contener la mejor combinación y reduce el número de combinaciones necesarias para encontrar buenos resultados.
Optimización evolutiva: Inspirada en algoritmos genéticos, utiliza conceptos como mutación, selección y cruce para explorar el espacio de búsqueda.
Ventajas: Puede encontrar combinaciones óptimas en espacios de búsqueda complejos.
Desventajas: Computacionalmente costoso y más difícil de implementar.

[] CAPÍTULO 10: TRABAJAR CON DATOS SIN ETIQUETAR (ANÁLISIS DE AGRUPACIÓN)

} 10.1 INTRODUCCIÓN

Técnicas de aprendizaje supervisado: Crean modelos de Aprendizaje Automático utilizando datos cuya respuesta ya se conocía (clasificación de etiquetas de clase en capacitación de datos).
Técnicas de aprendizaje no supervisado: Construyen modelos de Aprendizaje Automático que permiten descubrir estructuras ocultas en los datos donde no se sabe la respuesta correcta de antemano (agrupación de un modelo que intenta encontrar una relación natural de agrupar datos para que los elementos del mismo grupo sean más similares entre sí que con aquellos de diferentes grupos).

} 10.2 K-MEANS

K-means: Uno de los algoritmos de agrupamiento más populares utilizados tanto en el mundo académico como en la industria, el cual intenta encontrar grupos de objetos similares que estén más relacionados entre sí, exceptuando objetos de otros grupos (agrupación de documentos, música y películas por diferentes temas o la búsqueda de clientes que compartan intereses similares basados en comportamientos de compra comunes como base para la recomendación).
Agrupación basada en prototipos: Indica que cada grupo está representado por un prototipo, que suele ser el centroide de puntos similares con características continuas, o el medoide (minimiza la distancia a todos los demás puntos que pertenecen a un grupo particular) en el caso de los rasgos categóricos.
Ventajas: Extremadamente fácil de implementar y computacionalmente muy eficiente en comparación con otros algoritmos de agrupación, pertenece a la categoría de agrupamiento basado en prototipos donde existen otras categorías de agrupamiento, como el jerárquico y el agrupamiento basado en densidad, y es muy bueno para identificar grupos con una forma esférica, esto se debe a que el algoritmo minimiza la suma de la distancia de los cuadrados entre los puntos de datos y el centroide del grupo, en un espacio euclidiano esta minimización tiende a formar grupos que son esféricos alrededor de sus centroides.
Inconvenientes: Se especifica el k número de grupos a priori por lo que una inapropiada elección de k puede dar como resultado un rendimiento de agrupación deficiente.
Pasos: Paso 1 (elegir aleatoriamente k centroides de los ejemplos como grupo inicial), paso 2 (asignar cada ejemplo al centroide más cercano o similar, $\mu(j), j \in \{1, \dots, k\}$), paso 3 (mover los centroides al centro de los ejemplos asignados) y paso 4 (repetir los pasos 2 y 3 hasta que las asignaciones de clústeres no cambien la tolerancia de cambio definida por el usuario o el número máximo de iteraciones alcanzado).
Similitud entre objetos: Opuesto a la distancia (distancia euclidiana al cuadrado entre dos puntos x e y en un espacio de m-dimensional para agrupar objetos).
Escalado de funciones: Cuando se aplica k-means a datos del mundo real usando una métrica de distancia euclidiana, se quiere asegurar de que las características se miden en la misma escala para aplicar la estandarización de puntuación z o min-max escalando si es necesario.
Método del codo: Estima el número óptimo de grupos k para una tarea determinada, si k aumenta, la inercia disminuirá, ya que los ejemplos estarán más cerca de los centroides que se han asignado. La idea detrás del método del codo es identificar el valor de k donde la distorsión comienza a aumentar más rápidamente.
Análisis de silueta: Herramienta gráfica que traza una medida de qué tan estrechamente agrupados están los ejemplos en los grupos (qué tan similar es un objeto a su propio grupo en comparación con otros grupos).
Puntuación cercana a 1: Indica que el punto de datos está bien agrupado dentro de su propio grupo y está claramente separado de otros grupos, lo que sugiere que el punto de datos es similar a otros puntos en su grupo y diferente de puntos en grupos vecinos (cúmulo compacto y bien definido).
Puntuación de 0: Indica que el punto de datos está en el límite entre dos grupos, lo que significa que el punto de datos está equidistante entre su propio grupo y el punto de datos más cercano. En este caso, el punto podría pertenecer a cualquiera de los grupos, lo que sugiere que los racimos no están bien separados.
Puntuación cercana a -1: Indica que el punto de datos probablemente esté mal asignado a su grupo actual. Esto sugiere que el punto de datos es más similar a un grupo vecino que al cluster al que ha sido asignado. Es una señal de que el cluster está borroso o que los datos del punto podrían estar en el grupo equivocado.

} 10.3 ÁRBOLES JERÁRQUICOS

Agrupamiento jerárquico: Organiza los datos en una estructura jerárquica, similar a un árbol, donde cada nivel de la jerarquía representa una agrupación diferente de los datos. Esta estructura permite observar cómo se agrupan los datos en diferentes niveles de semejanza.
Ventajas: Permite trazar dendrogramas (visualizaciones de una jerarquía binaria de agrupamiento) que puede ayudar con la interpretación de los resultados y no se necesita especificar el número de grupos por adelantado.
Agrupamiento jerárquico divisivo: Se comienza con un grupo que abarca el conjunto de datos completo y se divide iterativamente en grupos más pequeños hasta que cada grupo solo contenga un ejemplo.
Agrupamiento jerárquico aglomerativo: Se toma lo opuesto, es decir, se comienza con cada ejemplo como un grupo individual y se fusionan los pares de clusters más cercanos hasta que solo quede un cluster.
Pasos: Paso 1 (calcular una matriz de distancias por pares de todos los ejemplos), paso 2 (representar cada punto de datos como un grupo único), paso 3 (fusionar los dos grupos más cercanos según la distancia entre los miembros más diferentes o distantes), paso 4 (actualizar la matriz de enlace del clúster) y paso 5 (repetir los pasos 2, 3 y 4 hasta que quede un solo grupo).
Enlace simple: Se calculan las distancias entre los miembros más similares o más cercanos para cada par de grupos y fusionar los dos grupos para los cuales la distancia entre los los miembros más cercanos son los más pequeños.
Enlace completo: Similar al del enlace simple pero, en lugar de comparar los miembros más cercanos en cada par de grupos, se comparan los miembros más diferentes (o más lejanos) realizando la fusión.
Matriz de distancias: Equivale a calcular la distancia euclidiana entre cada par de ejemplos de entrada en el conjunto de datos.
Matriz de enlace: Consta de varias filas donde cada fila representa una fusión. Las primeras dos columnas denotan los miembros más diferentes en cada grupo, la tercera columna informa la distancia entre esos miembros y la última columna devuelve el recuento de miembros de cada grupo.

} 10.4 DBSCAN

DBSCAN (Agrupación Espacial de Aplicaciones con Ruido Basada en Densidad): Asigna etiquetas basadas en regiones densas de puntos, donde la noción de densidad se define como el número de puntos dentro de un radio específico ε.
Criterios: Un punto se considera 'central' si al menos un número específico de puntos vecinos se encuentra dentro de un radio específico ε, un punto fronterizo es un punto que tiene menos vecinos dentro de ε, pero se encuentra dentro del radio ε de un punto central, y también se consideran todos los demás puntos que no son centrales ni puntos de ruido fronterizos.
Pasos: Paso 1 (formar un grupo separado para cada punto central o grupo conectado de núcleos, donde los puntos centrales están conectados si no están más lejos que ε) y paso 2 (asignar cada punto fronterizo al grupo de su punto central correspondiente).

[] CAPÍTULO 14: CLASIFICACIÓN DE IMÁGENES CON REDES NEURONALES CONVOLUCIONALES PROFUNDAS

} 14.1 REDES NEURONALES CONVOLUCIONALES (CNN)

Redes neuronales convolucionales (CNN): Familia de modelos que fueron originalmente inspirados en cómo funciona la corteza visual del cerebro humano cuando se reconocen objetos. Además, pueden aprender automáticamente características de datos sin procesar que son más útiles para una tarea particular.
Características: Las primeras capas (las que están inmediatamente después de la capa de entrada) extraen características de bajo nivel a partir de datos sin procesar, y las capas posteriores, a menudo capas completamente conectadas, como en un perceptrón multicapa (MLP), utiliza estas características para predecir un objetivo continuo (etiqueta de valor o clase). También construyen la llamada "jerarquía de características" combinando las características de bajo nivel en forma de capas para formar características de alto nivel (si se están tratando imágenes, entonces el nivel bajo de características, como bordes y manchas, se extraen de capas anteriores, que se combinan para formar entidades de alto nivel).
Conectividad escasa: Un único elemento en el mapa de características está conectado a una pequeña porción de píxeles (esto es muy diferente de conectarse a toda la imagen de entrada, como en los MLP).
Compartir parámetros: Se utilizan los mismos pesos para diferentes parches de la imagen de entrada.

} 14.2 CONVOLUCIÓN DISCRETA EN 1 DIMENSIÓN

Convolución discreta en 1 dimensión: Compuesta por dos vectores x y w se denota por y = x * w, en el cual el vector x es la nuestra de entrada (a veces llamada señal) y a w se le llama filtro o núcleo.
Modos de relleno: En modo completo, el parámetro de relleno p se establece en p = m – 1. Un relleno completo aumenta las dimensiones de la salida, aunque rara vez se usa en arquitecturas CNN. Generalmente se utiliza el mismo modo de relleno para garantizar que el vector de salida tiene el mismo tamaño que el vector de entrada x. En este caso, el parámetro de relleno p se calcula según el tamaño del filtro, junto con el requisito de que el tamaño de entrada y el tamaño de salida sean los mismos. Finalmente, calcular una convolución en modo válido hace referencia al caso en el que p = 0 (sin relleno).
Tamaño de salida de la convolución: Ttotal = [(Tvector + 2*relleno - Filtro) / zancada] + 1

} 14.3 CONVOLUCIÓN DISCRETA EN 2 DIMENSIONES

Convolución discreta en 2 dimensiones: Cuando se trabaja con entradas en dos dimensiones, como una matriz, $\boldsymbol{X}n1\times n2$, y la matriz de filtro, $\boldsymbol{W}m1\times m2$, donde $m1 \le n1$ y $m2 \le n2$, se tiene que la matriz $\boldsymbol{Y} = \boldsymbol{X} * \boldsymbol{W}$ es el resultado de una convolución en dos dimensiones entre $\boldsymbol{X}$ y $\boldsymbol{W}$.

} 14.4 SUBMUESTREO Y AGRUPACIÓN DE CAPAS

Submuestreo: Aplicado normalmente en dos formas de operaciones de agrupación en CNN (máxima y media). La capa de agrupación generalmente se denota por $Pn1\times n2$. Aquí, el subíndice determina el tamaño del vecindario (número de píxeles vecinos en cada dimensión) donde la operación máxima o media es realizada. A este tipo de vecindario se hace referencia como tamaño de la agrupación.
Agrupación de capas (max-pooling): Introduce una invariancia local. Esto significa que pequeños cambios en el vecindario local no cambian el resultado de la agrupación máxima. Por tanto, esto ayuda a generar características que sean más resistentes al ruido en la entrada de datos.
Características: La agrupación disminuye el tamaño de las características, lo que resulta en una mayor capacidad de eficiencia computacional. Además, reducir el número de funciones puede reducir también el grado de sobreajuste. Tradicionalmente, se supone que la agrupación no se superpone. La agrupación generalmente se realiza en vecindarios que no se superponen, lo que se puede hacer estableciendo el parámetro de zancada igual al tamaño de la agrupación. Por ejemplo, una capa de agrupación que no se superpone, $Pn1\times n2$, requiere un paso s = (n1, n2). Si bien la agrupación sigue siendo una parte esencial de muchas arquitecturas de CNN, también se han desarrollado varias arquitecturas CNN sin utilizar capas de agrupación. En lugar de utilizar capas de agrupación para reducir el tamaño de la entidad, los investigadores usan capas convolucionales con un paso de 2.

} 14.5 CANALES DE ENTRADA Y DE COLOR

Canales: Una entrada a una capa convolucional puede contener una o más matrices de dos dimensiones o matrices de dimensiones N1 × N2 (altura y ancho de la imagen en píxeles).

Implementaciones convencionales de capas convolucionales: Esperan una representación tensorial de rango 3 como entrada, por ejemplo, una matriz tridimensional $XN1×N2×Cin$, donde Cin es el número de canales de entrada. Por ejemplo, considerando imágenes como entrada a la primera capa de una CNN. Si la imagen está coloreada y utiliza el modo de color RGB, entonces Cin = 3 (para los canales de color rojo, verde y azul en RGB). Sin embargo, si la imagen está en escala de grises, entonces se tiene Cin = 1, porque solo hay un canal con los valores de intensidad de píxeles en escala de grises.

} 14.6 FUNCIONES DE ACTIVACIÓN

Funciones de activación (ReLU): Utilizadas principalmente en las capas intermedias u ocultas de una NN para agregar no linealidades al modelo.

Sigmoide (binario) y softmax (multiclase): Se agregan en la última capa (salida), lo que da como resultado probabilidades de membresía de clase como salida del modelo. Si las activaciones sigmoide o softmax no están incluidas en la capa de salida, entonces el modelo calculará los logits en lugar de las probabilidades de pertenencia a una clase.

} 14.7 FUNCIONES DE PÉRDIDA PARA CLASIFICACIÓN

Funciones de pérdida: Entropía cruzada binaria (función de pérdida utilizada en clasificación binaria con una sola unidad de salida) y la entropía cruzada categórica (función de pérdida utilizada para clasificación multiclase).

[] ANEXO: LARGE LANGUAGE MODELS (LLM)

} A.1 IA GENERATIVA

IA generativa: Término amplio que se puede utilizar para cualquier sistema de IA cuya función principal sea generar contenido como imágenes, texto, código, etc (generadores de imágenes como Midjourney o Stable Diffusion, generadores de texto como GPT-4, Claude o Llama y generadores de código como Github Copilot).

} A.2 ¿QUÉ ES UN MODELO DE LENGUAJE?

Modelo de lenguaje: Sistema de inteligencia artificial generativo basado en una máquina / modelo de aprendizaje que pretende generar un lenguaje plausible. Estos modelos funcionan estimando la probabilidad de que un token o secuencia de tokens que ocurren dentro de una secuencia más larga de tokens.

} A.3 ¿QUÉ ES UN LLM?

Modelos de lenguaje grandes (LLMs): Modelos de lenguaje que utilizan técnicas de aprendizaje profundo con un gran número de parámetros (desde millones hasta incluso billones). Estos modelos pueden capturar patrones complejos en el lenguaje y generar texto que a menudo es indistinguible de lo escrito por humanos.

} A.4 ¿QUÉ TAMAÑO TIENE UN LLM?

Parámetros: Pesos que el modelo de aprendizaje profundo aprendió durante el proceso de capacitación, donde los LLM tienen millones, miles de millones o incluso billones de parámetros (BERT con 110 millones de parámetros, PaLM 2 con 340 billones de parámetros y GPT-4 con 1 trillón de parámetros). En general, cuantos más parámetros tenga un LLM, más sofisticadas son las tareas que puede realizar.

} A.5 ¿CÓMO SE INTERACTÚA CON LOS LLM?

Características: El texto que se le pasa a un LLM se conoce como mensaje, el espacio que está disponible para el mensaje se llama ventana de contexto, y ésta suele ser lo suficientemente grande para unos pocos miles de palabras pero difiere de un modelo a otro. El resultado del modelo se llama finalización y el acto de utilizar el modelo para generar texto se conoce como inferencia.

} A.6 CASOS DE USO DE LLM

Predicción: Concepto básico detrás del texto LLM generado. Esta técnica se utiliza para una gran variedad de tareas en LLM (redacción de información, resumen, traducción y recuperación de información).

} A.7 ¿CÓMO FUNCIONAN LOS LLM?

Arquitectura del transformador: Escala de manera eficiente para utilizar una GPU de múltiples núcleos, procesa datos de entrada en paralelo haciendo uso de los conjuntos de datos de entrenamiento más grandes y es capaz de aprender a prestar atención al significado de las palabras que han sido procesadas.
Autoatención: Mecanismo utilizado para capturar dependencias y relaciones identificando y sopesando la importancia dentro de una secuencia de entrada.
Tokenización: Conversión de las palabras en números, donde cada número representa una posición en un diccionario (vocabulario).
Capa de incrustación: Espacio vectorial entrenable de alta dimensión donde cada token se representa como un vector multidimensional y ocupa una ubicación única dentro de ese espacio.
Información posicional: Se agrega explícitamente al modelo para retener la información sobre el orden de las palabras en una oración.
Codificación posicional: Esquema a través del cual se mantiene el orden del conocimiento de los objetos en una secuencia.
Capa de autoatención: El modelo analiza las relaciones entre los tokens en la secuencia de entrada y atiende a diferentes partes de la secuencia de entrada para capturar mejor las dependencias contextuales entre las palabras.

} A.8 GENERACIÓN DE TEXTO CON LLM

Proceso de predicción general: Primero, las palabras de entrada se tokenizan y estos tokens se insertan en la entrada en el lado del codificador de la red. Los tokens se pasan a través de la capa de incrustación y luego se introducen en las capas de atención de múltiples cabezas. Las salidas de las capas de atención de múltiples cabezas se alimentan a través de una red de avance a la salida del codificador. En este punto, los datos que salen del codificador son una representación profunda de la estructura y el significado de la secuencia de entrada. Esta representación se inserta en el medio del decodificador para influir en los mecanismos de autoatención del decodificador. A continuación, se agrega un token de inicio de secuencia a la entrada del decodificador. Esto hace que el decodificador prediga el siguiente token basándose en la comprensión contextual que proporciona el codificador. La salida de las capas de autoatención del decodificador pasan a través del red de avance del decodificador y a través de una capa de salida final softmax. En este punto, se obtiene la primera ficha. Este bucle continúa, pasando el token de salida nuevamente al decodificador de entrada para desencadenar la generación del siguiente token, hasta que el modelo prediga un token de fin de secuencia. En este punto, la secuencia final de tokens se puede destokenizar en palabras, obteniendo así la salida.

} A.9 VARIACIONES DE LA ARQUITECTURA LLM

Modelos de codificador-decodificador: Realizan tareas de secuencia a secuencia, tales como la traducción, donde la secuencia de entrada y la secuencia de salida pueden tener diferentes longitudes. Se pueden entrenar este tipo de modelos para realizar una generación de tareas de texto (BART y T5).
Modelos de solo codificador: Modelos de secuencia a secuencia (la secuencia de entrada y la secuencia de salida tienen la misma longitud) utilizados para extraer información relevante de la entrada y generar incrustaciones para otras tareas posteriores como clasificación o análisis (BERT).
Modelos de solo decodificador: Uno de los más utilizados en la actualidad. Son similares a los modelos codificador-decodificador, pero la información de entrada no está codificada explícitamente. Están entrenados para predecir el siguiente token en una secuencia dada la fichas anteriores y pueden generalizar la mayoría de las tareas (GPT y LlaMA).

} A.10 PROMPTING

Prompt engineering (prompting): Trabajo para desarrollar y mejorar el mensaje se conoce como mensaje de ingeniería. Una estrategia poderosa para lograr que el modelo produzca mejores resultados es incluir ejemplos de la tarea que desea que realice el modelo dentro del mensaje.
Aprendizaje en contexto: Ayuda a los LLM a aprender más sobre la tarea que se solicita al incluir ejemplos o datos adicionales y proporciona ejemplos dentro de la ventana de contexto.

} A.11 FINE-TUNING

Ajuste fino (fine-tuning): Proceso de aprendizaje supervisado donde se utiliza un conjunto de datos de ejemplos etiquetados para actualizar las ponderaciones del LLM.
Ajuste fino de instrucciones: Entrena el modelo utilizando ejemplos que demuestran cómo debe responder a una instrucción específica (los ejemplos etiquetados son pares de sugerencias para completar).
Desventajas: El proceso puede conducir a un fenómeno llamado olvido catastrófico. Esto sucede porque el proceso de ajuste completo modifica los pesos del LLM original. Si bien esto conduce a un gran rendimiento en una sola tarea de ajuste fino, puede degradar el rendimiento en otras tareas.
Ajuste de instrucciones multitarea: Ajuste de múltiples tareas al mismo tiempo.
Ajuste eficiente de parámetros (PEFT): Un conjunto de técnicas que preserva los pesos del entrenamiento LLM original, es decir, solo una pequeña cantidad de adaptadores específicos de capas y parámetros.
Métodos selectivos: Aquellos que afinan solo un subconjunto de parámetros del LLM original con varios enfoques para identificar qué parámetros se desea actualizar. Existe la opción de entrenar solo ciertos componentes del modelo o capas específicas, o incluso tipos de parámetros individuales.
Métodos de reparametrización: También funciona con los parámetros del LLM original, pero reduciendo el número de parámetros a entrenar mediante la creación de nuevos parámetros de rango bajo mediante transformaciones de los pesos originales de la red. Una técnica comúnmente utilizada de este tipo es la adaptación de bajo rango de modelos de lenguaje grandes (LoRA).
Métodos aditivos: Realizan ajustes manteniendo todos los pesos LLM originales congelados e introduciendo nuevos componentes entrenables.
Métodos adaptadores: Añaden nuevas capas entrenables a la arquitectura del modelo, normalmente dentro de los componentes codificadores-decodificadores después de las capas de atención o de retroalimentación.
Métodos de avisos suaves: Mantienen la arquitectura del modelo fija y congelada, y centrarse en manipular la entrada para lograr un mejor rendimiento. Esto se puede hacer añadiendo parámetros entrenables a las incorporaciones rápidas o mantener la entrada fija y volver a entrenar pesas incrustadas.

} A.12 LORA

LoRA: Congela todos los parámetros del modelo original y luego inyecta un par de matrices de descomposición de rangos junto con los pesos originales. Las dimensiones de las matrices más pequeñas se establecen de manera que su producto sea una matriz con las mismas dimensiones que los pesos que se están modificando.
Características: A modo de inferencia, las dos matrices de bajo rango se multiplican para obtener una matriz con las mismas dimensiones que los pesos congelados. Luego se agregan éstos a los pesos originales y se reemplazan en el modelo con estos valores actualizados. Ahora existe un modelo LoRA ajustado que puede llevar a cabo una tarea específica. Hay poco o ningún impacto en la latencia de inferencia.

} A.13 PROMPTING SUAVE

Prompting suave: Se agrega al incrustar vectores que representan el texto de entrada. Los vectores de aviso suave tienen la misma longitud que la incrustación de vectores de las fichas de lenguaje. Se puede entrenar un conjunto diferente de indicaciones suaves para cada tarea y luego cambiarlas fácilmente en el momento de la inferencia.

} A.14 PARÁMETROS DE CONFIGURACIÓN PARA INFERENCIA

Max new tokens: Utilizado para limitar el número de tokens que el modelo generará. Es el máximo de tokens nuevos, no una cantidad fija de tokens nuevos generados (si se alcanza una condición de parada, el modelo finaliza la generación).

Salida de la capa softmax del transformador: Probabilidad de distribución en todo el diccionario de palabras que el modelo usa.

Muestreo / Decodificación codicioso: Funciona muy bien para generaciones cortas, pero es susceptible a palabras repetidas o secuencias repetidas de palabras.

Muestreo aleatorio: Forma de introducir cierta variabilidad donde el modelo elige una palabra de salida en aleatorio utilizando la distribución de probabilidad para ponderar la selección. Al utilizar esta técnica, la probabilidad de que las palabras se repitan es reducida.

Top K: Limita las opciones y al mismo tiempo permite cierta variabilidad especificando que el modelo elige sólo entre los K tokens con mayor probabilidad utilizando la técnica de muestreo aleatorio. Una configuración de decodificación voraz es equivalente a Top K=1.

Top P: Limita el muestreo aleatorio a tokens cuyas probabilidades combinadas no excedan P. No se utiliza a menos que establezca el valor del parámetro Top P en algo distinto al valor predeterminado de 1.

Top K y Top P: Cuando se especifican ambos parámetros (K y P), Top K se aplica primero, y cualquier token por debajo del límite establecido por Top K se considera que tiene una probabilidad de cero cuando se calcula Top P.

Temperatura: Influye en la forma de la distribución de probabilidad de la capa de salida softmax del diccionario de palabras utilizado por el modelo. En general, cuanto mayor es la temperatura, mayor es la aleatoriedad y cuanto menor es la temperatura, menor es la aleatoriedad. El valor de temperatura es un factor de escala que se aplica a la distribución de probabilidad de la capa de salida softmax y afecta la forma de la distribución.