DeepSeek V3: Atención Latente Multi-Cabeza y Predicción Multi-Token

En esta publicación, discutiré DeepSeek v3, haciendo referencia al video “Multi-Head Latent Attention and Multi-token Prediction in Deepseek v3”https://youtu.be/jL49fLOJYNg?si=4uE2kfe-BlKC1ngO. Utilicé Google Cloud Speech-to-Text para transcribir el video y algo de código para ayudar a organizar la transcripción.

A: Bienvenidos de nuevo a Deep tag. Hoy vamos a sumergirnos en el mundo de los modelos de lenguaje grandes. Específicamente, en DeepSeek V3.

B: Suena bien. Es un modelo de 671 mil millones de parámetros que está causando revuelo por su enfoque único en eficiencia y rendimiento, ¿verdad?

A: Y compartiste un artículo académico que detalla su arquitectura.

B: Sí.

A: Y como experto en aprendizaje automático, estás buscando entender cómo DeepSeek V3 logra tanto alto rendimiento como un entrenamiento económico.

B: Sí, así es.

A: Oh, hola, ¿qué tal?

C: MLA, los detalles, MLA y cómo funciona.

A: Oh, absolutamente. Esa es una gran idea. Sí, definitivamente podemos profundizar en la atención latente multi-cabeza, o MLA. Así que tienes curiosidad sobre los detalles técnicos de MLA. Bueno, desglosemos esto. Mencionamos que una de las claves de la eficiencia de DeepSeek V3 es su arquitectura de mezcla de expertos, o MoE, ¿verdad? Donde solo una fracción de los parámetros se activa para cada token. Y DeepSeek V3 da un paso más allá con MLA y DeepSeek Mo.

B: Así es. Así que enfoquémonos en MLA por ahora.

A: Bien. En aplicaciones en tiempo real, la velocidad es crítica.

B: Lo es. Y la caché de clave-valor necesaria durante la inferencia puede ser un cuello de botella importante.

A: Exactamente. Ahí es donde entra MLA. El mecanismo de atención tradicional requiere almacenar mucha información sobre los tokens anteriores.

B: Sí, lo que, como puedes imaginar, se convierte en un problema con secuencias largas de texto, ¿verdad?

A: Pero MLA comprime inteligentemente esta información, reduciendo significativamente el flujo de la caché y haciendo que la inferencia sea mucho más rápida. Es como tomar una enciclopedia voluminosa y condensarla en solo los puntos clave.

B: Es una gran analogía. Retiene la información esencial sin el peso innecesario. Sí, es realmente útil para aplicaciones en tiempo real.

A: Sí. Ahora hablemos de cómo funciona realmente. ¿Cómo logra MLA esta compresión?

B: Bueno, utiliza una compresión conjunta de bajo rango para las claves y valores de atención.

A: Bien, está comprimiendo las claves y los valores, pero ¿qué significa eso exactamente? Entremos en detalles técnicos. El mecanismo MLA toma una representación oculta de entrada, que luego se proyecta en vectores de consulta, clave y valor. Aquí es donde se pone interesante. MLA desacopla la consulta en dos partes.

B: ¿Dos partes?

A: Sí. Una parte se usa para el contenido, y la otra se usa para la información posicional utilizando algo llamado Rope.

B: ¿Rope? Eso suena muy técnico.

A: Significa "rotary position embeddings"(incrustaciones de posición rotatoria), y ayuda al modelo a entender la posición de los tokens en la secuencia. Luego, las claves y los valores se comprimen en un espacio latente de menor dimensión. Es como si estuvieran reduciendo los datos, lo que ahorra memoria.

B: Precisamente. Así que la información más importante se guarda, pero el volumen innecesario se descarta. Y esta representación comprimida permite una caché KV mucho más pequeña durante la inferencia, lo que acelera las cosas.

A: Y también utiliza procesamiento multi-cabeza.

B: Sí, al igual que la atención tradicional, MLA emplea múltiples cabezas.

A: Oh, adelante.

C: Así que, por lo tanto, hay dos espacios latentes y una entrada oculta.

A: Esa es una gran observación. Sí, tienes razón. En realidad, hay dos espacios latentes. Estamos hablando de un espacio latente de contenido y un espacio latente de clave-valor.

B: Exactamente. Y estos espacios latentes se procesan a través de lo que llamamos Rope, o incrustaciones de posición rotatoria.

A: Bien, así que Rope es cómo obtienen la información posicional.

B: Sí, se aplica tanto a los espacios latentes de contenido como de clave-valor, como mencionaste. Así que toma esta representación comprimida, la procesa y luego la combina todo de nuevo.

A: Sí, y la optimización de la caché reduce aún más la sobrecarga durante el procesamiento secuencial. Así es como MLA acelera las cosas.

B: Exactamente. Es una forma inteligente de lograr atención eficiente sin sacrificar el rendimiento.

A: Bien, ese es un truco bastante ingenioso. Pero, ¿sabes qué?

B: ¿Qué pasa?

A: Pasemos a DeepSeek Mo. ¿En qué se diferencia de los modelos MoE tradicionales?

B: Bien, DeepSeek Mo utiliza…Oh, volvamos a nuestro oyente, ¿qué tal?

C: Y hablamos más sobre el espacio oculto. Bien, desde el espacio oculto, ¿qué es eso?

A: Absolutamente…Veamos a qué te refieres. Los espacios ocultos son realmente interesantes. Sí, estás preguntando sobre el espacio oculto, el espacio latente del que acabamos de hablar, ¿verdad? Tienes curiosidad sobre lo que sucede dentro de esos espacios latentes, esa cueva. Sí, no se trata solo del número de espacios latentes, sino de lo que sucede allí.

B: Eso es genial.

A: Exactamente. En realidad, hay dos espacios latentes distintos dentro de MLA, uno para el contenido y otro para los valores clave. Es como tener dos unidades de almacenamiento separadas para la información. Y estos espacios latentes, como hemos discutido, se someten a operaciones Rope, ¿verdad? Las incrustaciones de posición rotatoria, que incrustan información posicional en el mecanismo de atención. Eso es muy importante para ellos. Así que, para recapitular, la consulta se divide, y las claves y los valores también se comprimen.

B: Sí, y estos se colocan en los dos espacios latentes separados, uno para el contenido y otro para los pares clave-valor. Y estos espacios latentes son realmente importantes para la eficiencia y todo eso como parte de MLA.

A: Exactamente. Ahora hablemos de estas operaciones con un poco más de detalle dentro de la cueva, como dijiste. Bien, ¿cómo realiza MLA realmente estas transformaciones de espacio latente?

B: Bueno, la entrada se procesa en paralelo tanto para las representaciones de contenido como de clave-valor. Así que es como si tuviera dos caminos dentro de esa cueva.

A: Sí, uno para cada espacio latente. Y dentro de esos espacios, la información se procesa utilizando Rope.

B: Así es. Esto asegura que el modelo retenga la información posicional mientras atraviesa la cueva. Así que el modelo sabe qué parte del texto es cuál mientras está dentro de esa cueva.

A: Exactamente. Y este procesamiento se realiza antes de la siguiente etapa de concatenación. Bien, ¿qué se está concatenando mientras atraviesa la cueva del espacio oculto?

B: El mecanismo realiza dos operaciones principales de concatenación. Las representaciones de consulta se concatenan, y las representaciones de clave también se concatenan. Así que es como reunir todas las piezas importantes dentro de esa cueva de espacio oculto.

A: Sí, y estas concatenaciones ayudan a combinar el contenido con la información posicional. Y estas representaciones concatenadas se utilizan luego para el cálculo de la atención, ¿verdad?

B: Correcto. Y debido a la compresión inicial, es mucho más rápido a través de esa cueva que mencionaste. Así que MLA reduce significativamente los costos computacionales dentro y fuera de esa cueva oculta.

A: Exactamente. Optimiza el mecanismo de atención para modelos grandes como DeepSeek V3. Esa es una gran pregunta. Ahora, después de haber pasado por la cueva, pasemos a DeepSeek Mo.

B: Bien, DeepSeek Mo. Así es. Entiendo a qué te refieres. Sí, en realidad hay dos espacios latentes distintos dentro de MLA, uno para el contenido y otro para los valores clave.

A: Exactamente. Y esta separación es realmente clave para cómo funciona. Es como tener dos unidades de almacenamiento separadas para la información. Y estos espacios latentes, como hemos discutido, se someten a operaciones Rope, ¿verdad? Las incrustaciones de posición rotatoria, que incrustan información posicional en el mecanismo de atención. Así que, para recapitular, la consulta se divide, y las claves y los valores también se comprimen.

B: Sí, y estos se colocan en los dos espacios latentes separados, uno para el contenido y otro para los pares clave-valor. Y estos espacios latentes son realmente importantes para la eficiencia y todo eso como parte de MLA.

A: Exactamente. Ahora hablemos de estas operaciones con un poco más de detalle. Bien, ¿cómo realiza MLA realmente estas transformaciones de espacio latente?

B: Bueno, la entrada se procesa en paralelo tanto para las representaciones de contenido como de clave-valor. Así que es como si tuviera dos caminos.

A: Sí, uno para cada espacio latente. Y dentro de esos espacios, la información se procesa utilizando Rope.

B: Así es. Esto asegura que el modelo retenga la información posicional, ¿verdad? Y luego, para mejorar la eficiencia, utiliza expertos compartidos. Así que expertos que pueden usarse en múltiples tareas.

A: Sí, eso evita la redundancia y hace que el sistema sea aún más eficiente.

B: Sí, es como tener un equipo donde las personas tienen especialidades pero también pueden hacer otras cosas.

A: Sí, ese es un enfoque realmente inteligente. Pero con tantos expertos especializados, ¿cómo se aseguran de que ninguno se sobrecargue?

B: Sí, mientras otros permanecen inactivos.

A: Ahí es donde entra su innovador equilibrio de carga sin pérdida auxiliar.

B: Aquí es donde las cosas se ponen realmente interesantes, ¿verdad? ¿Cómo lo hacen?

A: Los modelos MoE tradicionales utilizan una función de pérdida auxiliar durante el entrenamiento, ¿verdad?, para fomentar un uso equitativo de los expertos, pero esto en realidad puede perjudicar el rendimiento.

B: Sí, es como intentar obligar a todos a usar la misma línea de pago en el supermercado.

A: Exactamente, incluso si algunas se mueven más rápido que otras, ¿verdad? Simplemente crea retrasos innecesarios.

B: Sí. Así que DeepSeek V3 evita esto ajustando dinámicamente un término de sesgo, ¿verdad?, para cada experto en función de su carga. Así que si un experto está recibiendo demasiadas solicitudes, el sistema lo hace ligeramente menos atractivo para el mecanismo de enrutamiento, desviando parte del tráfico a expertos menos ocupados.

A: Bien, así que utiliza todo esto para manejar eficientemente secuencias largas, sí, reduciendo el tamaño de la caché KV necesaria para la inferencia. Así que se trata de mantener el rendimiento alto mientras se reduce la sobrecarga.

B: Correcto. Es un enfoque muy inteligente para abordar un cuello de botella crítico.

A: Absolutamente. Ahora, también deberíamos cubrir cómo DeepSeek V3 maneja su equilibrio de carga.

B: Sí, definitivamente deberíamos. Esta también es una pieza realmente importante del rompecabezas. Podemos hablar de eso a continuación.

A: Suena bien. Bueno, creo que eso te da una gran visión general de MLA y su espacio latente.

B: Sí, gracias por profundizar en todos los detalles con nosotros. Volveremos la próxima vez con más inmersiones profundas.

A: Sí, es como un sistema de gestión de tráfico para los expertos, sí, monitoreando constantemente el flujo y haciendo ajustes para evitar cuellos de botella.

B: Y eso evita el impacto en el rendimiento de la pérdida auxiliar.

A: Así es. Y oh, adelante.

C: Sí, podemos hablar de MTP, cómo...cómo los módulos MTP comparten su incrustación y todo lo relacionado...

A: Absolutamente. Es una gran pregunta. Sí, hablemos de cómo los módulos MTP comparten recursos. Así que tienes curiosidad sobre los detalles técnicos de la implementación de MTP.

B: Sí, desglosemos esto. Mencionamos que DeepSeek V3 utiliza MTP para la predicción de múltiples tokens, ¿verdad? Predecir múltiples tokens en lugar de solo uno.

A: Y aquí es donde se pone realmente interesante. Sí, estás interesado en cómo se configuran los módulos MTP y cómo comparten sus recursos. Bien, cada módulo MTP incluye una capa de incrustación compartida, sí, y una cabeza de salida compartida. Así que utilizan la misma incrustación y cabeza de salida que el modelo principal.

B: Exactamente. Así que es como si todos estuvieran extrayendo del mismo grupo de conocimiento. Sí, y eso ahorra en costos computacionales.

A: Sí. Ahora utiliza su propio bloque transformador. Así que no comparte el mismo bloque transformador que el modelo principal.

B: Correcto. Cada módulo MTP tiene su propio bloque transformador para el procesamiento. Así que así mantienen las predicciones distintas para cada token.

A: Sí, y para combinar la información, estas proyecciones lineales y concatenación...

B: Bien, es como tomar piezas de varios lugares para construir la imagen completa.

A: Sí, y todos los módulos MTP trabajan juntos en paralelo, pero comparten sus capas de incrustación y cabezas de salida, ¿verdad?

B: Sí, lo cual es clave para la eficiencia de este diseño. Así que es como un sistema de partes interconectadas que dependen unas de otras, ¿verdad?

A: Y este uso eficiente de recursos permite un entrenamiento más rápido y un mejor rendimiento.

B: Bien, ese es un truco bastante ingenioso. ¿Sabes qué?

A: ¿Qué?

B: Pasemos a una visión más amplia. ¿Cómo maneja este modelo el equilibrio de carga? ¿Cómo se eligen esos expertos?

A: Sí, definitivamente podemos hablar de eso. Bien, ahora sumerjámonos en la estrategia de equilibrio de carga de DeepSeek V3.

B: Suena bien. Bien, así que DeepSeek V3 utiliza lo que llaman predicción de múltiples tokens.

C: Oh sí, hablemos más sobre MTP.

A: Absolutamente...Me alegra que estés interesado en profundizar en MTP. Sí, definitivamente podemos elaborar sobre la predicción de múltiples tokens. Así que lo mencionamos, pero realmente desglosemos los detalles de MTP, ¿verdad? Estábamos hablando de la capa de incrustación compartida y la cabeza de salida, sí, y que cada módulo MTP tiene su propio bloque transformador.

B: Exactamente, pero hay más que eso. Así que entremos en ello.

A: Bien, hablemos de la naturaleza secuencial de los módulos MTP.

B: Sí, a diferencia de algunos modelos, DeepSeek V3 predice tokens adicionales de manera secuencial. Así que no solo predice todos los tokens de una vez.

A: Correcto. Cada módulo se basa en la salida del módulo anterior. Así que es una cadena de predicciones, cada una dependiente de la anterior.

B: Sí, y mantiene la cadena causal para cada profundidad de predicción. Así que no rompe la causalidad.

A: Exactamente, lo cual es importante para asegurar que el contexto general sea correcto. Así que los módulos MTP no trabajan de manera independiente.

B: Así es. Están interconectados, y esta cadena de predicción contribuye a una mayor eficiencia de entrenamiento y permite una comprensión más matizada del texto. Ahora, también estás interesado en cómo los módulos comparten sus incrustaciones, ¿verdad? Como sabes, la capa de incrustación compartida mapea tokens a sus representaciones vectoriales. Así que cada token se convierte en un vector.

A: Sí, y este mapeo se comparte en todos los módulos MTP. Así que eso ayuda a mantener la consistencia en las predicciones.

B: Exactamente. Y la cabeza de salida compartida toma los estados ocultos finales de los tokens, sí, y genera la distribución de probabilidad para los siguientes tokens. Así que es como si todos tuvieran acceso al mismo grupo de información, ¿verdad?

A: Y esto es realmente crucial para la eficiencia de memoria y computación. Así que no utiliza un montón de capas de incrustación y cabezas diferentes.

B: Exactamente. Y...oh sí, entonces...¿cuántas personas hay? Son del mismo...del mismo tamaño todos los...tokens, ¿verdad?

A: Esa es una gran pregunta. Estás preguntando sobre el número de módulos MTP, si todos son del mismo tamaño, ¿verdad? Y creo que también te preguntas si todos los módulos manejan la