09 - ¡Red Neuronal Desde Cero estilo FRAMEWORK de DEEP LEARNING - Parte 2! - Transcript

En el vídeo anterior vimos cómo programar las clases que nos permiten instanciar un modelo de red normal profundo con un número de capas arbitrario.

En este vídeo vamos a programar la función softmax, así como la función de costo y el loop que nos va a permitir entrenar nuestro modelo.

Además a lo largo de este vídeo vamos a hacer un poquito de debugging para poder entrenar nuestro modelo.

Entonces continuamos.

Ya casi estamos listos, ahora vamos a utilizar la función de costo.

Es importante que notemos que también tenemos que implementar la función de softmax para la salida para la última capa de la red normal.

Entonces vamos a implementar una función que calcula las dos y esto lo vamos a hacer como una función.

Podríamos hacerlo como una clase, pero es común que se encuentre, que se pueda acceder a una implementación de una función de esta función de costo.

Por ejemplo en PyTorch existe la función cross entropy que también implementa la parte de softmax.

Entonces eso es lo que vamos a hacer en este caso.

Softmax cross entropy y lo que vamos a hacer es que va a recibir los scores y va a recibir las etiquetas.

Y para esto vamos a implementar exactamente la función softmax como lo implementamos en el vídeo anterior.

Estas funciones, esta función de softmax y esta función de cross entropy vamos a implementarlo.

Vamos a crear una variable que se llama batch size y esta va a ser igual a los elementos.

Recordemos que shape tiene en su dimensión de columnas tiene el número de elementos en ese batch.

Vamos a extraerlo y vamos a calcular los exponentes de los scores.

Esto ya lo hemos hecho anteriormente, voy a ir un poco rápido.

Vamos a calcular las probabilidades scores dividido entre x scores punto zoom y nos aseguramos que la axis sea igual a 0.

Esto lo hicimos más a detalle en el vídeo donde implementamos una función para nuestra red normal y ahora vamos a implementar la parte que corresponde a la función de costo.

Entonces esto sería la salida de nuestro softmax y vamos a crear una variable que se llama prets que va a ser una copia de props.

Esto es simplemente para poder regresarla porque ahorita vamos a modificar props.

Vamos a implementar la derivada de la función de costo.

Primero implementamos y hat que sería la predicción y esto estaría dado por pero sería props y recordemos hacíamos un y squeeze el número de elementos por el número de elementos en el batch size.

De esta forma lo que estamos haciendo es seleccionar los elementos correspondientes a la clase correcta de nuestra matriz que contiene las probabilidades y lo hacemos para todos los elementos que hay en el batch size.

Los guardamos en y hat y ahora sí calculamos el costo que estaría dado por la suma para todos los elementos del logaritmo natural de este valor y hat.

Lo que estamos haciendo aquí básicamente es menos el logaritmo natural de la predicción de nuestro modelo y dividimos entre el número de elementos y ahora esta sería la función de costo.

Vamos a ponerle aquí costo y ahora nos permite lo que vamos a hacer es calcular gradientes y para esto lo único que hacemos aquí vamos a modificar props por eso lo guardamos guardamos una copia en prets y aquí lo que vamos a hacer es que todos los elementos de props que corresponden a la clase correcta para todos los ejemplos se les va a restar un 1 que es el valor que tendría la clase correcta es decir vamos a seleccionar la predicción de nuestro modelo es esto mismo que hicimos acá la predicción de nuestro modelo pero lo vamos a modificar directamente en props y le vamos a restar el valor que correspondería a la clase correcta es decir ya que la clase correcta la la probabilidad para la clase correcta debe ser 1 entonces lo que estamos haciendo aquí es esa ecuación que ya habíamos visto que era que el gradiente era era igual a y hat - y esta parte nos permite seleccionar y hat es decir el elemento que está prediciendo nuestro modelo para todos los ejemplos del mini batch a ese número que está prediciendo le estamos restando uno es exactamente esta ecuación y prácticamente hemos terminado lo único que vamos a hacer es que voy a guardar los gradientes de x recordemos que x es la entrada x son los scores si nuestra función aquí que implementa softmax más la función de pérdida aquí recibe una variable z en este caso no le estamos llamando z le estamos llamando x y va a arrojar acá un valor de costo o de pérdida entonces nada más vamos a poner que el gradiente que corresponde a x porque x recordemos que va a ser un objeto que va a salir de la última capa es decir va a ser va ser un tipo z los que venimos viendo aquí sería un valor de z entonces lo que vamos a hacer es a ese valor de z que va a estar dado en esta variable x a su atributo grad le vamos a poner los valores de props porque esto ya es el gradiente de l es más podemos poner un comentario aquí este es el gradiente de l con respecto a x entonces va a ser props punto copi y sería todo lo único que vamos a hacer ahora es regresar las predicciones de nuestro modelo esto este en pretz y vamos a ponerle también el costo porque usualmente es lo que usamos con esta función no necesitamos regresar el gradiente porque ese ya lo estamos guardando directamente en el atributo grad del entrada y así tenemos la función de costo y ahora ya estamos terminando realmente lo único que necesitamos hacer es simplemente implementar nuestro loop de entrenamiento nuestra función de acura si y ejecutar nuestro modelo entonces vamos a hacer rápidamente nuestra definición de nuestro loop de entrenamiento ahora ya vamos a recibir nuestro modelo porque ya tenemos una clase que nos va a permitir crear un modelo vamos a recibir los hipops el número de hipops que queramos entrenar un número de elementos en un mi batch y también nuestro learning rate por default vamos a dejarlo como uno por diez a la menos tres y vamos a hacer nuevamente el loop de la misma manera que hicimos en este video simplemente que ahora ya no vamos a tener que actualizar todos los parámetros en range aquí vamos a ponerle el número de hipops y lo que vamos a hacer nuevamente vamos a crear nuestros mini batches por y si recordamos la función mini batches recibe el mb size mb de abajo size luego recibe x y luego recibe y entonces sería x serían los datos de entrenamiento x de abajo train coma y de bajo train y vamos a dejarle el true valor de fall nos falta un paréntesis y vamos a calcular los scores y para esto vamos a mandar llamar nuestro modelo que pasamos aquí y vemos que este modelo recordemos que es una instancia de sequential layers y lo que va a recibir es este x cuando lo mandemos a llamar simplemente como definimos la función guion bajo guion bajo col va a recibir estas entradas x entonces vamos a pasarle esos elementos x y recordemos que como veníamos haciendo desde nuestro nuestra implementación anterior calculamos cuando pasamos x los pasamos transpuestos ya quisimos la implementación de la capa lineal como la multiplicación directamente de w por x al transponerlos lo que nos va a permitir es tener en el número de filas las dimensiones de los de los elementos de entrada y en las columnas los ejemplos si vemos por ejemplo de que quis en bajo train punto shape vemos que tiene está en número de elementos dentro de elementos coma 784 que son los píxeles las imágenes los mini batches en este caso tendrían la misma forma simplemente nada más serían de 512 o 128 por 784 y al momento de hacer la transpuesta lo que vamos a tener es que sean de 784 por el número de elementos por 512 o 128 algo muy importante es que que pasemos x estos elementos x como un tipo de objeto en p guion bajo tensor esto es recordemos que lo pasamos así porque vamos a estar guardando el gradiente con respecto a esta variable entonces para poder guardar el gradiente vamos a ver que x por ejemplo va a guardar el gradiente de la función de pérdida con respecto a dicha x en el atributo grad por eso necesitamos pasar los elementos de x como un tipo de vista de la subclase del arreglo de non pi en p tensor ahora lo que vamos a hacer es calcular la función de costo si vemos la función de costo calcula regresa las predicciones y el costo en este caso no nos interesan las predicciones nada más vamos a calcular el costo y vamos a ponerlo como softmax el nombre de la función x entropy y lo que va a recibir es son los scores que acabamos de calcular y en este caso no necesitamos pasar y como un tipo en p tensor ya que no se va a calcular un gradiente con respecto a esta esta variable simplemente son las clases correctas que nos van a permitir calcular dicha función de costo y las predicciones y esta es la elegancia de nuestro modelo anteriormente mandábamos a llamar la función backward que hacía el backward para todos todos los elementos como los habíamos definido ahora simplemente vamos a pasar a hacer nuestro modelo punto backward y esto va a mandar a llamar esta función backward que a su vez va a mandar a llamar el backward de cada una de las capas en nuestro modelo de forma muy compacta esto esto realmente me parece muy muy elegante entonces así mandamos a llamar el backward de todos los elementos en nuestro modelo y finalmente hacemos el model punto update que esto es equivalente a un model punto step por ejemplo que implementaríamos en pytorch y el update si vemos nada más requiere el learning rate vamos a pasarle el learning rate con esto tenemos la función que nos va a permitir entrenar nuestro modelo vamos a imprimir cómo van las funciones de pérdida vamos a imprimir el costo costo dos puntos y vamos a imprimir aquí cost y vamos a imprimir también vamos a ponerle coma imprimamos también a curasim y estaría dado por vamos a mandar a llamar a la función de acurasi que todavía no hemos implementado acurasi pero la vamos a implementar de forma de forma muy similar a como implementamos anteriormente para recibir los valores de x de validación y el tamaño del minibatch si nos va a dar un error a ver si ahora sí sin embargo si mandamos a llamar este train pues no tenemos todavía definida la función de acurasi entonces vamos a definir la función acurasi que nos va a recibir x y y el mv size vamos a hacer un número de elementos un número de prevenciones correctas luego vamos a tener un total de elementos cero y vamos a hacer lo mismo por y vamos a recorrer a crear los minibatches que es realmente este mismo loop que estamos haciendo aquí coma x coma ya vamos a mandar a llamar ahora a nuestro modelo modelo y de nuevo a cuenta le vamos a pasar xd punto vamos a asumar los ejemplos que son correctos y para esto que vamos a hacer es que un empi empi son los elementos donde el máximo valor de las predicciones cuando esto sea igual al valor de la clase correcta entonces vamos a ir sumando este paréntesis va por acá porque cuando sean iguales con los elementos el mayor elemento de las predicciones se igual a la clase correcta recordemos que el índice en las predicciones va a corresponder al valor de la clase es decir el índice cero corresponde a un número cero el índice c5 corresponde a un número 5 cuando sean iguales lo que vamos a hacer es sumarlos todos estos o sea los verdaderos los va a tomar como un 1 y así vamos a sumar todos los elementos que son correctamente predichos y también vamos a sumar todos los elementos que llevamos hasta el momento y así calculamos las predicciones y ya estamos listos ahora sí terminamos de implementar de principio a fin todas las clases para crear nuestro modelo de retnernal ahora lo que vamos a hacer es nada más crear nuestro modelo instanciar nuestro modelo y vamos a llevarle al modelo un nombre muy original modelo y lo que vamos a hacer es mandar a llamar el objeto sequential va a ser de este tipo recordemos que nos recibe aquí las layers que queremos implementar y esto vamos a hacerlo es una lista que va a llevar linear recordemos que línea recibe el número de elementos de entrada en este caso sería 784 lo podríamos sacar directamente para para hacerlo más general pero en este caso vamos a dejarlo así 784 100 neuronas vamos a pasar otra una capa relo vamos a pasar otra capa linear que debe tener el número de elementos de 100 coma el número de elementos de salida por ejemplo 10 por ejemplo si ponemos aquí 200 estaríamos replicando básicamente el mismo modelo que implementamos previamente mb y en bajo size vamos a ponerlo de 512 el tamaño del mini batch learning rate que sea igual a 1 al menos 4 para hacerlo un poquito más pequeño y vamos a entrenar por 20 epochs y lo que vamos a hacer ahora es ya mandar a esta función mandar a llamar a esta función train a ver qué nos dice aquí nos dice que hay un error con el modelo donde se va a haber equivocado con el parámetro vamos a checarlo vamos a checar directamente sequential layers es en parámetros o sea es la función lineal es que puse input debería ser input size vemos nuestro modelo nos mal da un error nuevamente es en la línea 7 vamos a ver es un inicio te dice en pceros ok si aquí lo tenemos que pasar como tuple me hace falta me hace falta un par de paréntesis aquí y este cierra este este cierra este y me falta este ahora si esperemos que ahora si funcione ya nos crea el modelo bien y ahora si vamos a esperar que train funcione y recordemos recibimos el modelo que se llama modelo luego recibimos el número de hipops y pox luego vamos a mandarle envisage y leonín y si todo sale bien ahora deberíamos de ver como nuestra modelo entrena y hay algún error no importa dice aquí que hay un error en las dimensiones de las matrices en los scores ok ok ok aquí si cuando definimos nuestro modelo definimos 200 aquí y aquí simplemente 100 entonces me olvidé de cambiar aquí en 200 otro error no nos preocupemos vamos a ver update aquí fue un punto y no una coma vamos a línea 21 del modelo de la de la función update que debe ser una coma vamos a ver ahora parece que si no nos dice tuple index out of range total igual a ver que estamos haciendo acá por supuesto estoy mandando y debe ser uno de acuerdo estamos diciendo que queremos sumar el número de ejemplos que hay que me fui con la fin que le puse y lo veamos está entrenando y estamos llegando con un costo vemos como el costo va disminuyendo y la curious y va aumentando ok tuvimos algunos errores pequeñitos que pudimos resolver fácil pero estamos viendo que nuestro modelo está entrenando y esto está lo encuentro muy satisfactorio porque es una implementación muy elegante muy parecida a la forma en la que funcionan a la que se implementan los modelos en frameworks de deep learning vemos que alcanzamos básicamente 96.8 por ciento por ejemplo podríamos si quisiéramos incluir otra capa relo por ejemplo acá y luego nuestra otra capa de línea que sería de en este caso vamos a dejar esta de relo esta también vamos a dejarla de 200 200 otra capa lineal aquí que sería en este caso sería 200 por 10 y así cambiamos nuestro modelo y podríamos entrenar este modelo más complejo va a tardar un poquito más pero empiezan 92 93 95 96 va a tener una cura así muy parecido yo creo 97 alcanza 97 por ciento el ventilador de mi computadora se escucha ahora bastante fuerte 97.2 por ciento vamos a checar el accuracy pero para los valores de prueba los que no hemos visto en lo absoluto y así tendríamos el accuracy final que nuestro modelo arroja y así de test coma y bajo test verlo que pasó acá claro nos pide aquí así también lleva mini batch size y vamos a mandarle directamente mx mb que bajo así hacemos aquí así y directamente vemos que nos arroja 97.2 por ciento vamos a imprimir la cura así entonces tenemos 97.27 por ciento y finalmente si quisiéramos hacer algunas predicciones como hemos venido haciendo vamos a es más aquí no lo voy a escribir voy a copiar directamente de la libreta anterior lo hicimos en la libreta anterior entonces ya sabemos cómo hacerlo nada más para hacer algunas predicciones aleatorias y ahora lo queríamos nada más es que en vez de llamar a predict tendríamos que hacer model punto predict por plod number y lo que vamos a recibir es x test punto reshape esto está correcto el valor predicho es 8 7 8 vemos que nuestro modelo está funcionando bien está prediciendo de forma correcta bueno este quien sabe que sea es está interesante pero bueno puede ser un 6 podría ser uno podríamos poner también aquí quien que veamos el valor correcto el valor real es y ahora por ejemplo lo que veríamos sería simplemente white test white test de 9 2 0 1 con una cura así de 97 por ciento pues va a ser difícil que encontrar uno que nos equivo pero con esto terminamos hemos implementado una red neuronal multi kappa utilizando programación orientado objetos con con nuestras propias clases y esto nos permite entender muy bien cómo funciona nuestra red neuronal y ahora que vamos a utilizar frameworks de programación como paitos cuando veamos las funciones y cómo se implementan los modelos en paitos vamos a tener un entendimiento mucho mayor de cómo funciona paitos under the hood cómo funciona abajo nivel entonces en el siguiente vídeo continuaremos con este tipo de problemas utilizando paitos muchas gracias espero que este vídeo sea útil a lo largo de los últimos dos vídeos programamos la clase lineal la clase reloj la clase sequential así como la función de activación softmax y cross entropy de un modelo completo de red neuronal multi kappa esto es un logro muy importante ya que no solamente implementamos el modelo desde cero utilizando exclusivamente non pi y python sino también tenemos un modelo que es sumamente elegante que funciona de una manera similar a las a las clases como se implementan en frameworks de deep learning como paitos ya tenemos una idea muy clara de qué es lo que está pasando cómo instanciaron sus clases para poder hacer estas operaciones de forma muy muy eficiente y elegante obviamente no estoy diciendo que estas clases son equivalentes o son iguales a lo que hace paitos paitos por ejemplo tiene muchos más niveles de complejidad permite implementar prácticamente cualquier clase además de que nos permite que todo corren gpu sin embargo a nivel conceptual a nivel abstractión las operaciones son muy parecidas por lo tanto en el siguiente vídeo vamos a utilizar paitos para poder implementar un modelo de red neuronal multi kappa como hemos venido haciendo pero ahora de forma que maximicemos los beneficios de ahorro de tiempo y de poder computacional que paitos nos ofrece y por poder computacional me refiero a poder utilizar gpus en caso de que los tengamos disponibles continuamos y y y [MÚSICA]