Introducción

En lo que respecta al área de computación e informática, la interpretación de imágenes siempre a sido un problema difícil de resolver, en especial en el área de minería de datos. Sin embargo , los últimos avances han permitido detectar y comprender imágenes hasta el punto de automatizar un sistema de vigilancia comercial, que es usado alrededor del mundo, siendo capaces de reconocer personas y objetos en movimiento.

Pero el proceso ya es bastante difícil, ya que se debe extraer bastante información de las imágenes en bruto, debido a la enorme variedad de condiciones que pueden afectar a la extracción adecuada de los pixeles como información útil.

El problema que se pasara a resolver es la de la resolución automática del CAPTCHA, que son imágenes diseñadas para ser mas fácil de resolver para humanos pero difícil de resolver para un ordenador, impidiendo así el uso de programas automatizados usados en cuantas falsas o mensajes Spam.

Para conseguir este objetivo se llevara acabo el uso de redes neuronales, la creación propia de un conjunto de CAPTCHA y letras y las librerías scikit y PyBrain para el trabajo de imágenes y redes neuronales conjuntas en Python.

Todo este proceso se sostiene a base de las redes neuronales, siendo un algoritmo diseñado a base del funcionamiento del cerebro humano, compuesta de neuronas simples con una función simple de varias entradas que genera una única salida, calculando esa única salida en base a cálculos predefinidos en muchos casos. De forma individual no hacen mucho pero combinados son una fuerza poderosa de procesamiento muy útiles para el mundo de la minería de datos, que son incluso capaces de aprender de forma automática.

Desarrollo

Ya revisado de manera superficial el concepto de las redes neuronales, se procederá a la creación de un programa que sea capaz de que pueda vencer el CAPTCHA, nada complicado podría generarse usando solo letras. Para conseguirlo se deberá dividir la imagen en sus letras individuales, clasificar las letras, recombinar las letras para formar una nueva palabra y clasificar las palabras con un diccionario para tratar de corregir errores.

Utilizado la librerías antes mencionadas de Python, nuestro primer paso será dibujar una imagen y realizar un corte para almacenar los fragmentos en una matriz. El tamaño de adaptación para cada imagen que variara ligeramente con cada letra generada, para luego mostrarla como cualquier grafica con la librería matplotlib.

El paso siguiente seria identificar cada una de las letras separadas, usando funciones para encontrar conjuntos de pixeles que den el mismo valor, siendo las imágenes no identificadas valoradas con un cero, para ello se deben identificar las regiones del objeto para obtener la región actual de la imagen, tomando como parámetro el origen y el fin de la región a evaluar. Para finalmente extraer las subimagenes indexando la imagen aun arreglo simple para poder enseñar cada imagen fragmentada.

Luego generaremos una función con la cual podremos crear un conjunto de datos de letras, con diferentes valores para poder entrenar a la red neuronal, para reconocer cada letra. Pasando a la selección de valor de forma aleatoria desde la matriz que tiene las letras, se configurara para usar valores de corte al azar. Posteriormente creamos una función para mostrar el funcionamiento del conjunto de datos para luego devolver el valor de las letras junto con las imágenes, valores que oscilaran del 0 al 26, tratando de no exceder los valores que puede soportar cada neurona de la red neuronal se manejaran valores del 0 al 1 para identificar si la letra se parece o no.

La idea detrás del entrenamiento de las redes neuronales es que el clasificador pueda imitar su entorno, minimizando lo mas posible las diferencias, que será acelerado por la ejecución de una función de segmentación en el conjunto de datos de entrenamiento.

Normalmente el conjunto de datos seria tridimensional pero necesitaremos trabajar con un conjunto bidimensional, para construir un red neural que tomara una imagen como entrada y podrá predecir cada letra que se encuentre en la imagen.

Con todo esto realizado, estaremos listos par construir una red neural de 3 capas, 400 neuronas compondrán la primera capa y 26 neuronas estarán listas para la salida de datos, pero determinar el numero de neuronas en la capa oculta entre ambas, ya que demasiadas neuronas lo harían difícil de entrenar, por el momento se usaran 100 neuronas para la capa oculta.

Para corregir los posibles errores se usara el algoritmo de propagación inversa, que “culpara” a las neuronas por cada mala predicción, ajustando el peso hacia esas neuronas, este peso afectara a la taza de aprendizaje que será el parámetro del algoritmo (normalmente en un valor bajo). De forma concreta calcularemos la gradiente del error de la función, multiplicándola por la taza de aprendizaje y restándole el peso, y según el error y el peso que se este restando, este siempre tratara de corregirse a la dirección correcta, al menos hasta llegar a lo que se llamara optimo local (donde los pesos son similares pero el mejor conjunto de pesos). Este algoritmo retrocederá de capa en capa hasta llegar a la entrada de información, entonces los pesos de las conexiones neuronales serán actualizados.

El algoritmo de backdrop se ejecuta d manera iterativa usando los conjuntos de entrenamiento para ajustar el peso de las neuronas un poco, siendo factible el parar la ejecución cuando el error se reduce a una cantidad muy pequeña y pierde el sentido el entrenamiento.

Para evitar ese sentido de iteraciones, se definen “épocas” que determinaran la duración de la iteración del algoritmo, en este caso usaremos 20 épocas. Estas iteraciones darán como resultado una puntuación de 0.97, el cual es un gran resultado para una algoritmo neural tan simple como este.

La siguiente fase será la predicción de cada unos de los segmentos que se formaran del captcha dado, aceptando en la función el captcha, devolviendo la palabra predicha. Se ordenaran las subimagenes según el orden de ubicación, a su vez enviaremos datos a la primera capa de la red neuronal para contener la salida deseada, que constara de 26 números, cada uno relacionado con el índice de la letra predicha. Pero para la obtención del valor máximo de las salidas, se buscara en cada una de las letras y así subsecuentemente con la demás letras hasta forma la palabra predicha.

La utilización de un grafico, que demuestre la precisión de la predicción, se hace evidente para poder apreciar los valores de la frecuencia que demuestran cierta falta de precisión, que podremos solucionar con el uso de un diccionario. Que en vez de dar solo un predicción, también podremos comprobar si la palabra existe en el mismo, suponiendo que todas las palabras captcha estén en ingles esto no podría aplicarse de manera tan aleatoria. Uno de los problemas el la cantidad de variables a considerar para considerar una palabra similar, ya ser la extensión o viceversa.

Una solución a esta incógnita es el mecanismo Levenshtein para comparar 2 cadenas cortas y ver que tan similares son. Que consiste en ingresar la palabra en cualquier posición, eliminar una letra aleatoria y sustituirla por otra letra. El numero minimo de acciones para transformar la primera palabra en la segunda se da por la distancia, valores mas altos indican menor similaridad. Este método es excelente para probar errores de ortografía, de dictado o coincidencia en nombres. Con la creación de una distancia métrica, se le resta valor a la longitud de la palabra de predicción, para convertirlo en una métrica que indique la similitud entre palabras en los valores mas bajos.

Con esta mejoras implementadas, verificaremos si la palabra se encuentra dentro del diccionario, y en caso de no encontrar la palabra, el algoritmo devolverá la palabra mas cercana, calculando la distancia entre la palabra predicha y la palabra seleccionada en el diccionario, dando como resultado la palabra con mejor coincidencia. Esto mejora la precisión del algoritmo de manera muy considerable.

Conclusión.-

En conclusión podemos que el captcha generado no goza de la complejidad de los que se usan en los portales web, ya que solo se usaron 4 letras para palabras en ingles, sin embargo la maravilla de esta tecnología es que puede crecer para resolver problemas mas complejos, eso también gracias al potencial no explotado de las librerías utilizadas. A pesar de ello la precisión alcanzada en la mayoría de los casos de 0.97 %.

En resumen las redes neuronales son un conjunto de neuronas conectadas para realizar cálculos básicos de manera individual, pero juntos pueden llegar a resolver problemas complejos, ya poseen demasiado potencial para explotar.

**Universidad privada Franz Tamayo**

**Facultad de ingeniería**

**Carrera de ingeniería de sistemas**



**“Solucionador de CAPTCHA”**

Integrantes: Glenn Julián Castro Duarte, José Luis Rodríguez Peredo, Isaac Herrera Mareño, Diego Marca

Docente: Miguel Ángel Paco

Asignatura: Minería de datos

Cochabamba-Bolivia

2019