Capitulo 4

4.1 Entrenamiento del reconocedor CSLU Toolkit.

La documentación del CSLU Toolkit generada en el Center for Spoken Language Understanding, del Oregon Graduate Institute en conjunto con el laboratorio TLATOA de la UDLAP es la fuente oficial de información sobre los pasos a seguir. Este documento provee de información detallada de todo el proceso de entrenamiento y describe los pasos que se usan para entrenar un reconocedor basado en redes neuronales, utilizando los scripts desarrollados en Tlatoa.

El primer paso es organizar la estructura de directorios en tu máquina. El CSLU Toolkit asume una estructura definida para un corpus, la cual se debe mantener para garantizar un nivel de consistencia entre máquinas y que el código y los procedimientos se puedan compartir dentro de un grupo.

- El corpus reside en el path `c:\data\corpora\`
- El archivo que describe a todos los corpus es `c:\data\corpora\corpora`. Aquí esta un ejemplo del archivo corpora
- Cada corpus incluye tres subdirectorios: `speechfiles`, `transcriptions` y `files`.
- Dentro del subdirectorio `speechfiles` existe un subdirectorio por cada locutor en donde se encuentran los archivos de audio con extensión `.wav`
- Dentro del subdirectorio `transcriptions` existe un subdirectorio para cada locutor en donde se encuentran las transcripciones a nivel de texto, a nivel de palabra y a nivel fonético con extensión `.txt`, `.wrd`, y `.phn` respectivamente.
- Los reconocedores residen en `c:\data\recognizers\`
- Los subdirectorios del `c:\data\recognizers` se nombran de acuerdo al corpus. Por ejemplo, `c:\data\recognizers\adultosjulio` contiene reconocedores entrenados con el corpus de alfanuméricos.
- Cada experimento que se realiza con un corpus determinado debe hacerse creando un nuevo reconocedor. Por ejemplo, `c:\data\recognizers\adultosjulio\adultosjulio1` es el sistema base del corpus de letras.

- El CSLU Toolkit debe encontrarse en el path `C:\CSLU\`

Los siguientes pasos describen el procedimiento para crear un reconocedor en una máquina. Se asume que ya se ha etiquetado el corpus y está listo para ser usado en el entrenamiento.

Cada script crea un archivo con extensión .readme el cual describe con más detalle los pasos que se llevan a cabo.

I. Crear el archivo .vocab y .parts. El primero contiene el vocabulario del corpus y el archivo .parts específica el número de partes en que dividirán los fonemas.

En el apéndice E podemos ver un ejemplo del archivo *prueba1.vocab* y en el apéndice F podemos ver un ejemplo de un archivo *prueba1.parts.*

II. Crear la estructura de archivos para un reconocedor:

tclsh80 C:/CSLU/Toolkit/2.0/script/scripts/make_recognizer.tcl -name <*nombre del reconocedor*> -corpus <*nombre del corpus*> -sampling_rate <*frecuencia de muestreo de las grabaciones*>

Con este script se generan los componentes y la estructura de archivos para poder entrenar el reconocedor. Después de la etiqueta -name se coloca el nombre que uno le quiere dar al reconocedor y por lo tanto a la carpeta donde se guardarán todos los archivos, producto del entrenamiento. Los archivos .vocab que se pude ver el en el apéndice D y . parts que se encuentra en el apéndice E tendrán que llevar el mismo nombre que le daremos a nuestro reconocedor.

El script make_recognizer.tcl ejecuta otros scripts que son los siguientes:

- find_files.tcl: este encuentra los archivos utilizados para el entrenamiento, desarrollo y prueba del reconocedor.
- categories.tcl: genera las categorías que se entrenarán por medio de los archivos con extensiones .vocab, .parts e. info. También es creado el archivo .desc que contiene una descripción del reconocedor para que sea usado por los demás

scripts posteriormente, y también se genera un archivo .olddesc que contiene la misma información pero en un formato mas viejo.

Al momento en que se ejecutan estos scripts se hace una copia del .vocab y .parts en la carpeta de entrenamiento del reconocedor.

III. Generar los archivos con extensión .cat**:**

tclsh80 C:/CSLU/Toolkit/2.0/script/scripts/generate_cat_files.tcl -name <*nombre del reconocedor*> -corpus <*nombre del corpus*> -files <*nombre del corpus*>

Con este script se crea un nuevo directorio llamado “train” en el cual se generan archivos de categorías alineadas en el tiempo de la transcripción de texto o de transcripción fonética, estos archivos son generados con extensión .cat.

También se crea un archivo con extensión .train.counts con todas las categorías que contiene nuestro reconocedor de voz en las cuales se especifica la duración y las veces que se encuentran en las grabaciones. En el caso de las categorías que no se encuentran dentro del corpus de voz, las podemos ir relacionando con las categorías existentes *(create ties).*

Después de generar todos los archivos .cat, el script llama a "revise_desc.tcl" para atar categorías que tengan pocas muestras. Esta parte es interactiva y se le pide al usuario que responda a una serie de preguntas. En caso de que no se desee crear "ties" o revisar la información de las duraciones, sólo responde que “no” a las preguntas y se guardan los cambios.

IV. Generar los vectores de características a partir de los archivos de sonido que se usarán para entrenar:

tclsh80 C:/CSLU/Toolkit/2.0/script/scripts/gen_data.tcl -name <*nombre del reconocedor*> -corpus <*nombre del corpus*>

En este paso se crean los vectores de características a partir de los archivos de sonido en formato binario que se usarán para el entrenamiento de la red neuronal. Este script utiliza otros script como son:

- Pickfames.tcl: este script es para checar que frame va con cual categoría, también crea un archivo .pick que será usado mas adelante
- Genvec.tcl: genera vectores de cada frame que va ser entrenado.

V. Entrenar y probar las redes neuronales**:**

tclsh80 C:/CSLU/Toolkit/2.0/script/scripts/train_and_test_nets.tcl -name <*nombre del reconocedor*> -corpus <*nombre del corpus*>

Entrena y prueba una red neuronal en 30 iteraciones, guardando el resultado de las últimas 15 iteraciones. Al finalizar evalúan estas 15 redes y se selecciona la mejor de ellas.

Estos pasos pueden llegar a ser tardados por la cantidad de datos que contiene nuestro corpus, pero entre mas grabaciones tengamos es mejor la red neuronal por que contiene un mayor número de fonemas.

VI. Checar los errores usando browse.tcl y speechview.tcl:

tclsh c:/cslu/Toolkit/2.0/script/training_1.0/browse.tcl <*nombre del reconocedor*>.dev. <*nombre del corpus*>.files c:/cslu/Toolkit/2.0/script/training_1.0/recog.tcl <*la mejor red*> <*nombre del reconocedor*>.vocab <*nombre del reconocedor*>.train.olddesc -a wrdalign_<*la mejor red*>

```
wish80 c:/cslu/Toolkit/2.0/script/sview_1.0/speechview.tcl -Wf temp.wav -S -Lf temp.wrd -Lf temp.phn -update
```

Nota: Se ejecuta el primer comando desde un shell y el segundo en otro. En este caso, se utiliza la red que obtuvo mejores resultados, y en el archivo wrdalign_<*la*

mejor red> se encuentran los resultados de la evaluación de esta red neuronal con el conjunto de datos de desarrollo (<*nombre del reconocedor*>.dev. <*nombre del corpus*>.files). Los archivos temporales automáticamente son generados por el script browse.tcl, y el argumento -update del script speechview.tcl asegura que siempre que se generen nuevos temp.files, Speechview los cargará para desplegarlos.

4.2 Resultados del entrenamiento.

Se podría argumentar que el reconocimiento no es alto sin embargo esto se debe a que el vocabulario es muy extenso, que no se aplicaron técnicas para reducir el error o vocabularios de contextos específicos.

El principal objetivo de este entrenamiento fue detectar errores y depurar el corpus Tlatoa Adult Microphone Speech los errores que se detectaron fueron mal etiquetado a nivel palabra, fonema por ejemplo: la transcripción de la “ñ” que en ocasiones se transcribió como “ñ”,”nj”,”n~”, también había etiquetas que no eran correctas y señalaban sonidos que no pertenecen al habla como son: “snift, Unk, PAU”

Los resultados que se obtuvieron después de correr por cerca de 36 horas el entrenador de voz con todas las grabaciones fue la siguiente:

Entrenamiento con habla espontánea:

Itr	#Snt	#Words	Sub%	Ins%	Del%	WrdAcc%	SntCorr
30	325	1119	58.27%	41.20%	2.86%	-2.32%	36.62%
29	***325***	***1119***	***57.82%***	***37.09%***	***3.84%***	***1.25%***	***36.62%***
28	325	1119	61.30%	39.14%	2.50%	-2.95%	36.00%
27	325	1119	59.34%	41.82%	2.77%	-3.93%	36.00%
26	325	1119	60.86%	38.34%	2.95%	-2.14%	36.92%
25	325	1119	59.43%	40.93%	2.23%	-2.59%	34.15%
24	325	1119	59.70%	38.61%	2.77%	-1.07%	36.62%
23	325	1119	59.16%	40.93%	2.14%	-2.23%	37.85%
22	325	1119	59.25%	41.20%	2.95%	-3.40%	36.31%
21	325	1119	61.75%	40.75%	2.23%	-4.74%	36.31%
20	325	1119	60.41%	39.68%	2.41%	-2.50%	36.62%
19	325	1119	59.96%	36.73%	3.49%	-0.18%	35.69%

18	325	1119	60.05%	38.70%	2.77%	-1.52%	36.00%
17	325	1119	59.96%	40.13%	2.59%	-2.68%	33.85%
16	325	1119	60.05%	40.13%	2.14%	-2.32%	35.38%
15	325	1119	61.13%	37.53%	3.13%	-1.79%	35.38%

Best results (1.25, 36.62) with network nnet.29

Aquí nos muestra que el mejor resultado que se obtuvo fue el de la red numero 29, esto con todos los archivos que se grabaron de los 52 locutores en la tabla de abajo el mejor resultado fue la red 24 pero en esta se elimino el muestreo del habla espontánea.

Entrenamiento con habla espontánea:

Itr	#Snt	#Words	Sub%	Ins%	Del%	WrdAcc%	SntCorr
30	318	880	56.25%	39.55%	2.50%	1.70%	38.36%
29	318	880	56.59%	39.89%	1.82%	1.70%	38.99%
28	318	880	56.48%	43.75%	1.25%	-1.48%	39.31%
27	318	880	55.23%	42.05%	1.93%	0.80%	38.05%
26	318	880	56.70%	43.41%	1.36%	-1.48%	37.42%
25	318	880	57.39%	40.68%	1.48%	0.45%	39.31%
24	***318***	***880***	***56.36%***	***40.00%***	***1.82%***	***1.82%***	***38.36%***
23	318	880	57.16%	42.61%	1.70%	-1.48%	36.79%
22	318	880	58.64%	41.82%	1.59%	-2.05%	38.05%
21	318	880	57.05%	42.39%	2.16%	-1.59%	38.36%
20	318	880	56.02%	46.36%	1.82%	-4.20%	38.36%
19	318	880	57.05%	43.86%	1.93%	-2.84%	36.48%
18	318	880	57.16%	46.14%	1.25%	-4.55%	38.99%
17	318	880	54.20%	43.41%	2.61%	-0.23%	35.85%
16	318	880	56.48%	46.02%	2.27%	-4.77%	36.48%
15	318	880	54.89%	42.84%	2.05%	0.23%	38.68%

Best results (1.82, 38.36) with network nnet.24

Para poder ver como fue el reconocimiento del entrenamiento verificamos el archivo wrdalign_nnet en el cual todas las palabras que logro reconocer están en minúsculas y las palabras que no se logaron reconocer están en mayúsculas.

Lo se presenta abajo es parte de un archivo que registra las palabras que logro reconocer la mejor red del reconocedor de voz, lo que esta con letras minúsculas es lo que si logro reconocer lo que esta con letras mayúsculas no lo logro reconocer y del mismo modo lo que esta con signo de gatos puede ser ruido que detecto y palabras que no están bien

pronunciadas, el primer renglón es lo que esta en la grabación y lo del segundo renglón es lo que el reconocedor pudo igualar.

```
Cero #### ### PRIMERO de enero DEL dos mil
Cero ENTE MAR ORO     de enero QUE dos mil
```

RESUMEN

En este capitulo se habla de cómo se hizo un entrenamiento de reconocimiento de voz con ayuda de CSLU Toolkit esto para depurar de una mejor manera el corpus calidad micrófono, y se comparan las tablas de las dos mejores redes neuronales una con habla espontánea y la otra si habla espontánea, se explica que es cada una de las columnas de esta misma tabla y por que es la mejor red neuronal.

También se habla de las dificultades que se presentaron al momento en que el corpus se sometió al entrenamiento, como fueron errores en el etiquetado de las grabaciones y conflicto en el reconocimiento de la letra Ñ.