

Named Entity Recognition (NER GUI UNIX) manual de usuario.

Aplicaciones del Software.

El software de reconocimiento de entidades nombradas (NER), implementa técnicas de inteligencia artificial para extraer información relevante de un documento de forma automática. Es por esto por lo que usa tecnologías de última generación para desarrollar un framework que permita utilizarlas y presentarlas de una forma amigable al usuario.

El software permite encontrar entidades dentro de un texto de forma automática, de este modo agiliza y automatiza la tarea de identificar Personas, Lugares, Documentos, etc. El texto de interés puede ser una frase específica o un documento completo, de esta forma brinda versatilidad para sus campos de uso. Toda la información de las entidades encontradas en un documento se guardará en otro documento que podrá ser usado posteriormente para el análisis o procesamiento de información relevante.

El software cuenta con un modelo por defecto que está definido dentro del contexto comercial de la cámara de comercio de Colombia. De igual manera permite realizar el entrenamiento de nuevos modelos para otros contextos de información diferente, así se genera el modelo para su posterior uso.

Así el software agiliza la implementación de técnicas de procesamiento de lenguaje natural para el análisis de documentos.

Índice.

Sección	Página
1. Como instalar	2
2. Descripción del Menú	3
2.1. Tagger	3
2.1.1. Sentence	3
2.1.2. Document	5
2.2. Trainer	7
3. Como usarlo	10
3.1. Predicción por frase	10
3.2. Predicción por documento	11
3.3. Entrenamiento	13
4. Formatos	16
4.1. Documento de entrada	16
4.2. Documento de salida	16
4.3. CONLL-03	19
4.4. PRATECH	19
5. Solución de errores	22
6. Glosario	22

Cómo instalar

Para su instalación es necesario contar con Docker dentro del servidor o computador en el cual se instalará.

Desde el directorio principal donde se encuentra el Dockerfile, se debe ejecutar las siguientes líneas de comando en la consola del sistema:

- `docker build -t ner .`
- `docker run -it --name ner_container -p 8080:8080 --mount type=bind,source="$(pwd)"/data,target=/workspace/data ner`
- Ejecutar en el contenedor el comando `python execute_GUI.py`

Luego desde el navegador se debe acceder al siguiente enlace: <http://localhost:8080>.

Descripción del Menú

El menú principal del programa cuenta con 2 secciones *Tagger* y *Trainer*. En la sección *Tagger* se puede encontrar funcionalidades para la aplicación de un modelo pre-entrenado en el reconocimiento de entidades nombradas en frases o documentos. En la sección *Trainer* se puede encontrar el modo de entrenamiento de nuevos modelos para su posterior uso en la sección *Tagger*.

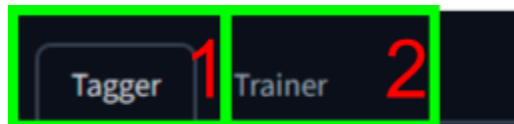


Figura 1. Menú principal.

A continuación, se describe a detalle cada parte de las secciones mencionadas.

Tagger

En esta sección se puede hacer el uso de modelos pre-entrenados en diferentes contextos para el reconocimiento de entidades nombradas.

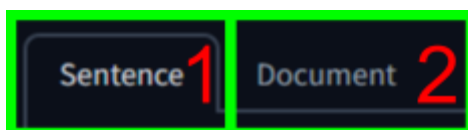


Figura 2. Menú del Tagger.

Esta sección tiene la posibilidad de aplicar el reconocimiento de entidades nombradas en una frase en texto plano en la subsección *Sentence*, o en un documento completo en formato JSON en la subsección *Document*.

Sentence

En la subsección *Sentence* se puede aplicar el reconocimiento de entidades nombradas en una frase dada.



Figura 3. Sentence interfaz gráfica.

Como se observa en la **Figura 3** esta subsección está compuesta de 3 partes: (1) El menú de preferencias del *Tagger* para frases, (2) La ventana de salida en donde se mostrará la frase etiquetada con las entidades reconocidas, y (3) la ventana de ejemplos donde se podrá escoger uno de 2 ejemplos propuestos.

A su vez el menú de configuraciones está compuesto de 4 subsecciones mostradas en la **Figura 4** en donde cada una controla una preferencia para el *Tagger* en frases.



Figura 4. Menú de configuraciones de Sentence.

Las preferencias son las siguientes:

1. **Model:** En este apartado se podrá seleccionar uno de los modelos existentes para realizar el reconocimiento de entidades nombradas, por defecto el programa cuenta con el modelo CCC.
2. **Sentence:** Este cuadro de texto está reservado para escribir la frase a la cual se le quiere aplicar el reconocimiento de entidades nombradas.
3. **CUDA:** Esta configuración permite indicarle al sistema el uso de una GPU si es posible. Si está en *true* el sistema buscará las GPU disponibles y hará uso de la primera encontrada, si no encuentra hará uso de la CPU. Si está en *false* el sistema usará la CPU directamente.
4. **Tag:** Este botón acciona el sistema, una vez los anteriores parámetros fueron dados, el sistema realiza el reconocimiento de entidades nombradas usando el modelo escogido, en la frase dada y la posibilidad de uso de GPU.

Document

En la subsección *Document* se puede aplicar el reconocimiento de entidades nombradas en un documento en formato JSON. Esta sección también genera un documento formato JSON donde se despliega la información de las entidades reconocidas en el documento.

Nota: Los formatos de los archivos de entrada y salida están descritos en la sección **formatos**.



Figura 5. Document interfaz gráfica.

Como se observa en la **Figura 5** esta subsección está compuesta de 3 partes: (1) El menú de preferencias del *Tagger* para documentos, (2) La ventana de salida en donde se mostrará el texto del documento ingresado etiquetado con las entidades reconocidas, y (3) la ventana de salida donde se muestra el contenido del documento JSON generado con información acerca de las entidades reconocidas.

A su vez el menú de configuraciones está compuesto de 5 subsecciones mostradas en la **Figura 6** en donde cada una controla una preferencia para el *Tagger* en Documentos.

The image shows a configuration menu for a document processing application. It is divided into five horizontal sections, each highlighted with a red number in the top right corner:

- 1 Model:** Contains three radio buttons labeled "CCC", "CCC_fast", and "CCC_new".
- 2 Input data file:** A large text area with the text "Coloque el archivo aquí", "- O -", and "Haga click para cargar".
- 3 Output data file path:** A text input field with the placeholder text "Enter path here...".
- 4 CUDA:** Contains two radio buttons labeled "true" and "false", with "false" being selected.
- 5 Tag:** A single text input field.

Figura 6. Menú de configuraciones de Document.

Las preferencias son las siguientes:

1. **Model:** En este apartado se podrá seleccionar uno de los modelos existentes para realizar el reconocimiento de entidades nombradas, por defecto el programa cuenta con el modelo CCC.
2. **Input data file:** Este cuadro de texto está reservado para cargar el documento en formato JSON al cuál se le desea aplicar el reconocimiento de entidades nombradas.
Nota: Formato definido en la sección **formatos**
3. **Output data file path:** Este cuadro de texto está reservado para escribir la ruta absoluta en la cual se desea guardar el documento resultante de aplicar el reconocimiento de entidades nombradas en un documento dado.

Nota: Formato de documento de salida definido en la sección **formato**.

4. **CUDA:** Esta configuración permite indicarle al sistema el uso de una GPU si es posible. Si está en *true* el sistema buscará las GPU disponibles y hará uso de la primera encontrada, si no encuentra hará uso de la CPU. Si está en *false* el sistema usará la CPU directamente.
5. **Tag:** Este botón acciona el sistema, una vez los anteriores parámetros fueron dados, el sistema realiza el reconocimiento de entidades nombradas usando el modelo escogido, en el documento dado, y guarda los resultados en la ruta de archivo dada, y analiza la posibilidad de uso de GPU.

Trainer

En esta sección se puede entrenar un nuevo modelo para el reconocimiento de entidades nombradas usando una base de datos en formato CONLL-03 o en formato PRATECH.

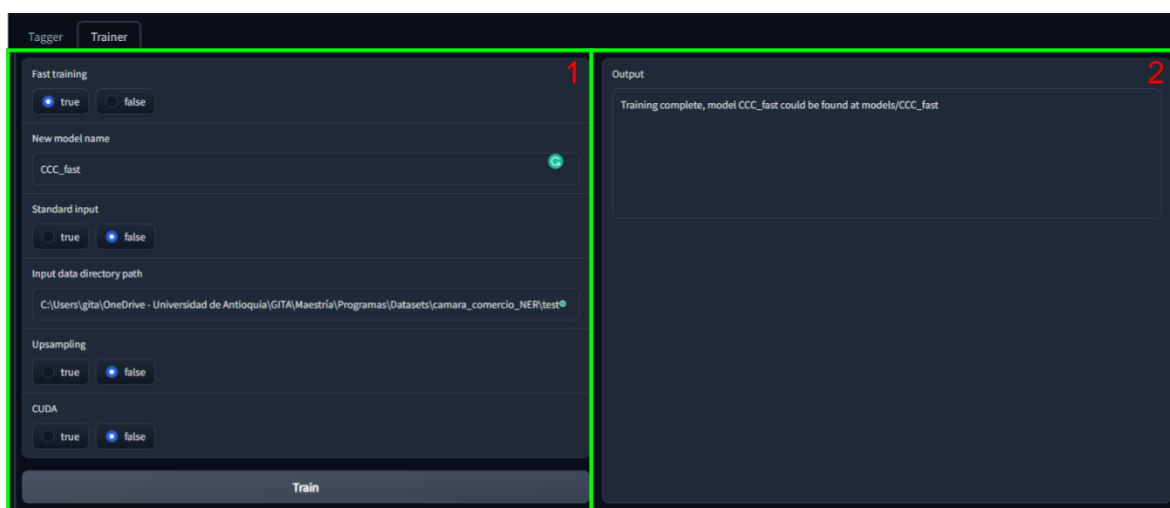


Figura 7. Trainer interfaz gráfica.

Como se observa en la **Figura 7** esta subsección está compuesta de 2 partes: (1) El menú de preferencias del *Trainer*, (2) La ventana de salida en donde se mostrará el estado del entrenamiento.

A su vez el menú de configuraciones está compuesto de 7 subsecciones mostradas en la **Figura 8** en donde cada una controla una preferencia para el *Trainer*.

The image shows a configuration menu for a Trainer interface. It consists of seven distinct sections, each with a red number in the top right corner:

- 1. Fast training:** A section with two radio buttons. The 'true' button is selected (indicated by a blue dot).
- 2. New model name:** A text input field with the placeholder text 'Enter model name here...'.
- 3. Standard input:** A section with two radio buttons. The 'false' button is selected (indicated by a blue dot).
- 4. Input data directory path:** A text input field with the placeholder text 'Enter path here...'.
- 5. Upsampling:** A section with two radio buttons. The 'false' button is selected (indicated by a blue dot).
- 6. CUDA:** A section with two radio buttons. The 'false' button is selected (indicated by a blue dot).
- 7. Train:** A large, dark button labeled 'Train' at the bottom of the menu.

Figura 8. Menú de configuraciones de Trainer.

Las preferencias son las siguientes:

1. **Fast Training:** Está configuración permite indicarle al sistema si se desea hacer un entrenamiento de prueba (rápido) o un entrenamiento completo. Si está en true hará un entrenamiento rápido para probar el sistema de entrenamiento con la base de datos dada, si está en false hará un entrenamiento completo. Esta opción es exclusivamente para testear el funcionamiento con una nueva base de datos, el modelo resultante del entrenamiento rápido no será de utilidad real. **Nota: Si se usa el nombre de un modelo existente este se sobrescribirá)**
2. **New model name:** En este campo se le indicará el nombre del nuevo modelo a generar.
3. **Standard input:** Esta opción le indica al sistema si los datos de entrenamiento están en formato CONLL-03 (estándar) o en formato PRATECH.
Nota: Formato definido en la sección *formatos*.
4. **Input data directory path:** Este cuadro de texto está reservado para escribir la ruta absoluta del directorio o carpeta en donde se encuentran los archivos para el entrenamiento de un nuevo modelo.
Nota: Formato definido en la sección *formatos*.
5. **Upsampling:** Está configuración permite indicarle al sistema si se desea hacer un aumento de los datos con técnicas de aumento enfocadas al reconocimiento de entidades

nombradas. Si está en *true* hará uso de las técnicas, si está en *false* no hará uso de las técnicas.

6. **CUDA:** Esta configuración permite indicarle al sistema el uso de una GPU si es posible. Si está en *true* el sistema buscará las GPU disponibles y hará uso de la primera encontrada, si no encuentra hará uso de la CPU. Si está en *false* el sistema usará la CPU directamente.
7. **Tag:** Este botón acciona el sistema, una vez los anteriores parámetros fueron dados, el sistema realiza el entrenamiento del nuevo modelo de reconocimiento de entidades nombradas usando el nombre dado, el tipo de datos de entrada que se encuentran en la ruta indicada, con la posibilidad de realizar aumento de datos y analiza la posibilidad de uso de GPU.

Como usarlo

Predicción por frase:

Para realizar un reconocimiento de entidades nombradas de una frase, primero hay que entrar al menú *Tagger*, en la sección *Sentence*. Una vez allí se debe seguir los siguientes pasos:

- (1) Seleccionar el modelo de interés de acuerdo con el contexto de las entidades deseadas.
- (2) Escribir la frase a la cuál se le quiere realizar el reconocimiento de entidades nombradas.
- (3) Seleccionar si se desea usar una GPU o no.
- (4) Hacer clic en el botón **Tag** el cual comenzará el proceso de reconocimiento de entidades nombradas.

Un ejemplo de esta configuración se muestra en la **Figura 9**.

Sentence Document

Model

☒ CCC ☐ CCC_fast ☐ CCC_new

Sentence

Camara de comercio de medellín. El ciudadano JAIME JARAMILLO VELEZ identificado con C.C. 12546987 ingresó al plantel el día 1/01/2022

CUDA

☒ true ☐ false

Tag

Figura 9. Predicción por frase.

En la ventana *Examples* se encuentran 2 ejemplos predefinidos para realizar el reconocimiento de entidades nombradas, como se muestra en la **Figura 10**.

Examples	
Model	Sentence
CCC	Camara de comercio de medellín. El ciudadano JAIME JARAMILLO VELEZ identificado con C.C. 12546987 ingresó al plantel el día 1/01/2022
CCC	Razón Social GASEOSAS GLACIAR S.A.S, ACTIVIDAD PRINCIPAL fabricación y distribución de bebidas endulzadas

Figura 10. Ejemplos de frase.

Una vez se configuren los campos y se accione el botón **Tag**, se podrá ver el estado en la pantalla de salida. Al final del proceso se visualizará el resultado en la ventana de salida. Un ejemplo de una frase luego del reconocimiento de entidades es mostrado en la **Figura 11**.



Figura 11. Frase con entidades reconocidas.

Si en el proceso de reconocimiento se presentó algún error podremos ver el código de error en la ventana de salida. Con este código podremos identificar el error y posibles soluciones en el capítulo de ***solución de errores***.

Predicción por documento:

Para realizar un reconocimiento de entidades nombradas de un documento, primero hay que entrar al menú *Tagger*, en la sección *Document*. Una vez allí se debe seguir los siguientes pasos:

- (1) Seleccionar el modelo de interés de acuerdo con el contexto de las entidades deseadas.
- (2) Arrastrar o seleccionar el documento al quien se le quiere realizar el reconocimiento de entidades nombradas.
- (3) Escribir la ruta absoluta donde se ubicará el archivo de salida.
- (4) Seleccionar si se desea usar una GPU o no.
- (5) Hacer clic en el botón **Tag** el cual comenzará el proceso de reconocimiento de entidades nombradas.

Un ejemplo de esta configuración se muestra en la **Figura 12**.

The image shows a web interface for document prediction. It has two tabs at the top: "Sentence" and "Document", with "Document" being the active tab. The interface is divided into five sections, each marked with a red number:

- 1**: The "Model" section, which contains three radio buttons: "CCC" (selected), "CCC_fast", and "CCC_new".
- 2**: The "Input data file" section, which shows a file named "0a61f100-85f6-11e8-86f2-47a419..." with a size of "256.6 KB" and a "Download" link.
- 3**: The "Output data file path" section, which contains a text input field with the path "C:\Users\gita\Desktop\output_file.json" and a green circular icon with a "G" on the right.
- 4**: The "CUDA" section, which contains two radio buttons: "true" (selected) and "false".
- 5**: A large button labeled "Tag" at the bottom of the interface.

Figura 12. Predicción por documento.

Una vez se configuren los campos y se accione el botón Tag, se podrá ver el estado en la pantalla de salida de texto (1). Al final del proceso se visualizará el resultado en la ventana de salida de texto (1) y el documento de salida en la ventana de salida de JSON (2). Un ejemplo de una frase luego del reconocimiento de entidades es mostrado en la **Figura 13**.



Figura 13. Texto y documento con entidades reconocidas.

Si en el proceso de reconocimiento se presentó algún error podremos ver el código de error en la ventana de salida. Con este código podremos identificar el error y posibles soluciones en el capítulo de ***solución de errores***.

Entrenamiento:

El entrenamiento tiene 2 modos, modo de prueba y modo completo: (1) El modo de prueba es un entrenamiento rápido en el cual se busca testear una nueva base de datos realizando una inspección de todo el proceso de entrenamiento sin un ajuste adecuado del modelo. (2) El modo completo es un entrenamiento donde se realiza todo el proceso de entrenamiento con un ajuste completo del modelo a la nueva base de datos. Este modo se selecciona en el campo *Fast Training* de la sección de configuraciones del *Trainer*.

Para realizar un entrenamiento de un nuevo modelo de reconocimiento de entidades nombradas, primero hay que entrar al menú *Trainer*. Una vez allí se debe seguir los siguientes pasos:

- (1) Seleccionar modo de prueba (true) o modo completo (false).
- (2) Escribir el nombre del nuevo modelo. **Nota: Si se usa el nombre de un modelo existente este se sobrescribirá**
- (3) Elegir si la entrada es en formato CONLL-03 (true) o en formato PRACTECH (false).
- (4) Escribir la ruta absoluta del directorio donde se encuentran los archivos de la base de datos para el entrenamiento.
- (5) Seleccionar si se desea aplicar técnicas de sobre muestreo a la nueva base de datos.
- (6) Seleccionar si se desea usar una GPU o no.

(7) Hacer clic en el botón **Train** el cual comenzará el proceso de entrenamiento de un nuevo modelo para el reconocimiento de entidades nombradas.

Un ejemplo de esta configuración se muestra en la **Figura 14**.

The image shows a dark-themed configuration window for training a model. It contains several sections, each with a red number in the top right corner indicating a step in the process:

- 1** **Fast training**: Two radio buttons, 'true' (selected) and 'false'.
- 2** **New model name**: A text input field with the placeholder 'Enter model name here...'.
- 3** **Standard input**: Two radio buttons, 'true' and 'false' (selected).
- 4** **Input data directory path**: A text input field with the placeholder 'Enter path here...'.
- 5** **Upsampling**: Two radio buttons, 'true' and 'false' (selected).
- 6** **CUDA**: Two radio buttons, 'true' and 'false' (selected).
- 7** **Train**: A large button at the bottom of the configuration panel.

Figura 14. Entrenamiento de un nuevo modelo.

Una vez se configuren los campos y se accione el botón Train, se podrá ver el estado en la pantalla de salida. Al final del proceso se visualizará el resultado en la ventana de salida de texto. Un ejemplo de la salida de un entrenamiento se muestra en la **Figura 15**.

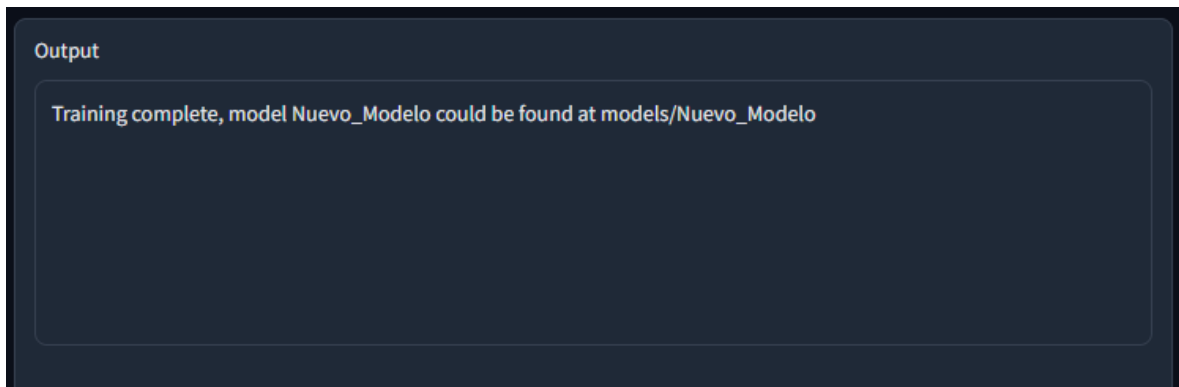


Figura 15. Entrenamiento de un nuevo modelo.

Si en el proceso de reconocimiento se presentó algún error podremos ver el código de error en la ventana de salida. Con este código podremos identificar el error y posibles soluciones en el capítulo de ***solución de errores***.

Nota: si se desea obtener el archivo del nuevo modelo se debe mover el archivo a la carpeta *data*.

Formatos.

Para el *Tagger* en documentos:

Documento de entrada.

El formato de entrada debe contener un objeto cuyo nombre “sentences” y su valor es una lista de bloques donde cada bloque tiene un objeto que lleva por nombre “text” y su valor es el texto de la frase.

En la **Figura 16** se muestra un ejemplo del formato de entrada

```
{
  sentences: [
    0: {
      text: "Frase ejemplo"
    },
    1: {
      text: "Frase ejemplo"
    }
  ]
}
```

Figura 16. Formato documento de entrada

Documento de salida.

El documento de salida cuenta con 4 objetos de nombres: text, text_labeled, sentences y entities.

- El objeto **text**, tiene por valor el texto plano del documento.
- El objeto **text_labeled** tiene por valor el texto plano etiquetado de la siguiente manera. Si un token del texto tiene una etiqueta diferente a “O”, el token se encierra en comillas y se le adiciona un ‘/' seguido de la etiqueta correspondiente.
- El objeto **sentences** tiene por valor una lista de bloques, cada bloque de la lista hace referencia a una frase del documento y está compuesto de los siguientes 3 objetos:
 - **text**: tiene por valor el texto plano de la frase
 - **text_labeled**: tiene por valor el texto plano etiquetado de la misma forma que el objeto text_labeled del documento.
 - **Tokens**: tiene por valor una lista de bloques, cada bloque de la lista hace referencia a un token de la frase y está compuesto por: (1) un objeto de nombre “text” que contiene por valor el texto plano del token y (2) un objeto de nombre label que contiene por valor la etiqueta de dicho token.
- El objeto **entities** tiene por valor una lista de bloques, donde cada bloque se refiere a cada mención de cada entidad encontrada en el documento, está compuesto de los siguientes objetos:

- **entity:** tiene por valor el nombre de la entidad.
- **index:** tiene por valor la posición a nivel de token dentro del texto plano del token al cuál la entidad se refiere.
- **word:** tiene por valor el texto del token.
- **start:** tiene por valor la posición a nivel de carácter del primer carácter del token
- **end:** tiene por valor la posición a nivel de carácter del último carácter del token

En la **Figura 17** se encuentra un ejemplo del formato de salida.

```

{
  text: "Frase ejemplo Frase ejemplo ",
  text_labeled: " "Frase"/Entity_Type ejemplo "Frase"/Entity_Type ejemplo ",
  sentences: [
    0: {
      text: "Frase ejemplo",
      text_labeled: " "Frase"/Entity_Type ejemplo",
      tokens: [
        0: {
          text: "Frase",
          label: "Entity_Type"
        },
        1: {
          text: "ejemplo",
          label: "0"
        }
      ]
    }
  ],
  1: {
    text: "Frase ejemplo",
    text_labeled: " "Frase"/Entity_Type ejemplo",
    tokens: [
      0: {
        text: "Frase",
        label: "Entity_Type"
      },
      1: {
        text: "ejemplo",
        label: "0"
      }
    ]
  }
],
  entities: [
    0: {
      entity: "Entity_Type",
      index: 0,
      word: "Frase",
      start: 0,
      end: 5
    },
    1: {
      entity: "Entity_Type",
      index: 2,
      word: "Frase",
      start: 14,
      end: 19
    }
  ]
}

```

Figura 17. Formato documento de salida

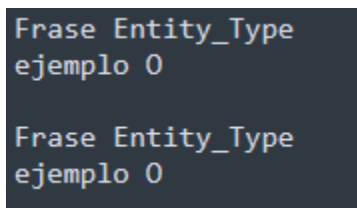
Para el Trainer:

CONLL-03

El formato CONLL-03 consiste en un documento de texto donde se encuentran diversas frases. Cada frase está compuesta por diversos tokens. Para el formato cada línea del documento corresponde a un token y su respectiva entidad separados por un espacio. La separación entre frases está dada por un salto de línea.

Nota: Para el uso de este formato el directorio de entrada debe contener 2 archivos: (1) *Train.txt*, el cual contendrá los datos de entrenamiento y (2) *Test.txt* el cual contendrá los datos de prueba. Ambos archivos deben estar en formato CONLL-03.

En la **Figura 18** se muestra un ejemplo de un archivo en formato CONLL-03.



```
Frase Entity_Type
ejemplo 0

Frase Entity_Type
ejemplo 0
```

Figura 18. Formato CONLL-03

PRATECH

El formato PRACTECH cuenta con 3 objetos de nombres: text, sentences y mentions.

- El objeto **text**, tiene por valor el texto plano del documento.
- El objeto **sentences** tiene por valor una lista de bloques, cada bloque de la lista hace referencia a una frase del documento y está compuesto de los siguientes 3 objetos:
 - **text**: tiene por valor el texto plano de la frase
 - **id**: tiene por valor un identificador de la posición de la frase dentro del documento, de la forma *s#*.
 - **Tokens**: tiene por valor una lista de bloques, cada bloque de la lista hace referencia a un token de la frase y está compuesto por: (1) un objeto de nombre "text" que contiene por valor el texto plano del token, (2) un objeto de nombre begin que contiene por valor la posición a nivel de carácter del primer carácter del token y (3) un objeto de nombre end que contiene por valor la posición a nivel de carácter del último carácter del token.
- El objeto **mentions** tiene por valor una lista de bloques, donde cada bloque se refiere a cada mención de cada entidad encontrada en el documento, está compuesto de los siguientes objetos:

- **id:** tiene por valor un identificador del número de frase y el número de mención a la que hace referencia., de la forma s#-m#.
- **type:** tiene por valor el nombre de la entidad.
- **word:** tiene por valor el texto del token.
- **begin:** tiene por valor la posición a nivel de carácter del primer carácter del token
- **end:** tiene por valor la posición a nivel de carácter del último carácter del token

Nota: Para el uso de este formato el directorio de entrada debe contener solo documentos JSON en el formato PRATECH y debe contener al menos 2.

En la **Figura 19** se encuentra un ejemplo del formato de salida.

```

{
  text: "Frase ejemplo Frase ejemplo",
  sentences: [
    0: {
      text: "Frase ejemplo",
      id: "s0",
      tokens: [
        0: {
          text: "Frase",
          begin: 0,
          end: 5
        },
        1: {
          text: "ejemplo",
          begin: 6,
          end: 13
        }
      ]
    },
    1: {
      text: "Frase ejemplo",
      id: "s1",
      tokens: [
        0: {
          text: "Frase",
          begin: 14,
          end: 19
        },
        1: {
          text: "ejemplo",
          begin: 20,
          end: 27
        }
      ]
    }
  ],
  mentions: [
    0: {
      id: "s0-m0",
      type: "Entity_type",
      begin: 0,
      end: 5
    },
    1: {
      id: "s1-m0",
      type: "Entity_type",
      begin: 14,
      end: 19
    }
  ]
}

```

Figura 19. Formato PRATECH

Solución de errores.

En la siguiente tabla se encuentra la información de los posibles errores, con su código de identificación, su causa y su posible solución

Código	Causa	Solución
1	Error cargando el modelo	Reentrenar el modelo verificando su completa finalización.
2	Error cargando el documento de entrada para predicción.	Revisar el documento de entrada ya que este puede estar vacío o corrupto.
3	Error en el formato del documento de entrada para predicción.	Revisar el formato del documento de entrada ya que no corresponde al formato que acepta el software.
4	Carpeta de documentos para entrenamiento vacía.	Usar una carpeta en donde se encuentren los documentos para el entrenamiento.
5	Error cargando el modelo base desde la nube para el entrenamiento.	Revisar la conexión a internet y reiniciar el entrenamiento.
7	Error en el entrenamiento del modelo.	Revisar la conexión a internet y reiniciar el entrenamiento.
8	Error en los archivos de entrada para el entrenamiento.	Revisar el formato de cada uno de los documentos de entrada para el entrenamiento.
9	La ruta de entrada para predicción de documento no corresponde a un archivo.	Verificar que la ruta existe y finalice en un archivo JSON.
10	El modelo seleccionado no existe.	Seleccionar un modelo existente.
11	Error en el documento de salida para predicción.	Verificar que la ruta existe y finalice en un archivo JSON.

Tabla 1. Resumen de errores.

Glosario.

NER: Reconocimiento de entidades nombradas. NER por sus siglas en inglés (Named Entity Recognition).

CPU: Unidad central de procesamiento. CPU por sus siglas en inglés (Central Processing Unit).

GPU: Unidad de procesamiento gráfico. GPU por sus siglas en inglés (Graphics Processing Unit).

JSON: Notación de objeto de JavaScript es un formato de texto sencillo para el intercambio de datos. JSON por sus siglas en inglés (JavaScript Object Notation).

Token: Unidad mínima de información de texto. En este contexto se refiere a una palabra.

Framework: es una estructura conceptual y tecnológica de asistencia definida, normalmente, con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software.