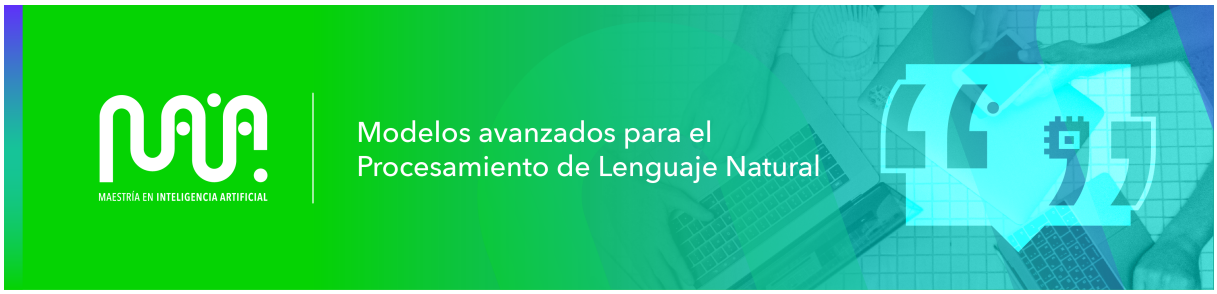

Instrucciones de Competencia Semana 8

Curso Modelos Avanzados de PLN

1 Introducción

Bienvenido a la competencia final del curso **Modelos Avanzados de PLN**. En este desafío, los participantes deben desarrollar un sistema de **canonicalización de pedidos de compra** con el objetivo de facilitar la implementación de **automatizaciones empresariales**. El propósito es diseñar un modelo capaz de interpretar y normalizar la información contenida en distintos formatos de pedidos, unificando su representación para optimizar procesos posteriores de análisis y procesamiento automatizado.

2 Resultados de Aprendizaje

Con esta actividad se busca que el estudiante pueda poner en práctica su capacidad de adaptar LLMs con fines específicos. Tras realizar esta actividad, se espera que el estudiante esté en la capacidad de:

1. Descargar y utilizar grandes modelos de lenguaje disponibles en Hugging Face.
2. Implementar técnicas de procesamiento de texto y otras estrategias para preparar un conjunto de datos de entrenamiento con el propósito de dotar a un modelo de lenguaje de ciertas capacidades o conocimientos.
3. Implementar diferentes estrategias de *finetuning* con el objetivo de seguir instrucciones y adquirir conocimiento de un dominio específico.
4. Usar modelos de lenguaje de tipo decoder para construir un sistema de canonicalización.

3 Descripción de la Competencia

Ustedes han sido contratados por una compañía de ventas de insumos para el público. Sus clientes esperan poder hacer sus pedidos por medio de correos electrónicos, mensajes de texto y documentos en formato `Markdown`. Sin embargo, resulta que esta compañía no tiene suficiente personal para procesar todas estas peticiones. Por esto se los ha contratado para crear un sistema de inteligencia artificial que sea capaz de convertir las órdenes de compra en lenguaje natural a un formato con el siguiente esquema **JSON**:

```
{
  "buyer": {
    "name": "string",
    "email": "string",
    "contact": {
      "phone": "string",
```

```
      "alt_email": "string",
      "preferred_contact": "email | phone | none"
    },
    "addresses": [
      {
        "street": "string",
        "city": "string",
        "state": "string",
        "postal_code": "string",
        "country": "US | CA | GB | ES | CO | DE | FR"
      }
    ]
  },
  "purchases": [
    {
      "product_name": "string",
      "quantity": "integer",
      "currency": "USD | EUR | GBP",
      "discount_code": "string"
    }
  ],
  "shipping": {
    "method": "standard | express | pickup",
    "preferred_by": "datetime"
  }
}
```

Aquí se presenta un ejemplo de una orden de compra en el formato esperado:

```
{
  "buyer": {
    "name": "Alice Johnson",
    "email": "alice@example.com",
    "contact": {
      "phone": "+1-555-234-5678",
      "alt_email": "ajohnson.alt@example.com",
      "preferred_contact": "email"
    },
    "addresses": [
      {
        "street": "123 Main St",
        "city": "Springfield",
        "state": "IL",
        "postal_code": "62704",
        "country": "US"
      }
    ]
  },
  "purchases": [
    {
      "product_name": "Wireless Keyboard",
      "quantity": 2,
      "currency": "USD",
      "discount_code": "WELCOME10"
    },
    {
      "product_name": "Laptop Stand",
      "quantity": 1
    }
```

```
  ],
  "shipping": {
    "method": "express",
    "preferred_by": "2025-11-10T15:00:00Z"
  }
}
```

El objetivo es utilizar un modelo decoder que reciba como entrada un texto de una orden de compra en lenguaje natural y lo transforme al esquema JSON. No hay garantía de que el texto de entrada contenga toda la información solicitada en el schema.

Para desarrollar este sistema deberán hacer *fine-tuning* a Qwen/Qwen3-0.6B-Base[1]. No se permite el uso de ningún otro modelo.

3.1 Evaluación

Para la evaluación se calcula el F1 score sobre los paths explícitos en los json. A continuación se explica su cálculo.

Descomposición del JSON en rutas ("paths")

Cada JSON (el esperado y el predicho) se descompone en una lista de *rutas* que representan todos los elementos dentro de él.

Por ejemplo, el siguiente JSON:

```
{
  "buyer": {"name": "Laura", "email": null},
  "purchases": [{"product_name": "Cafe", "units": 2}]
}
```

se convierte en una colección de rutas como:

Tipo	Ruta (path)
key	["buyer"]
key	["buyer", "name"]
field	["Laura", "buyer", "name"]
key	["buyer", "email"]
field	[null, "buyer", "email"]
key	["purchases"]
key	["purchases", "[]"]
key	["purchases", "[]", "product_name"]
field	["Café", "purchases", "[]", "product_name"]
key	["purchases", "[]", "units"]
field	[2, "purchases", "[]", "units"]

- **Las rutas de tipo key** representan las llaves o estructura del JSON.
- **Las rutas de tipo field** representan los valores concretos (como "Laura" o 2).

Comparación entre el JSON esperado y el predicho

- Una vez que ambos JSON se transforman en listas de rutas:
- Las rutas que aparecen en ambos se consideran **True Positives (TP)**.
- Las rutas que aparecen solo en el JSON predicho son **False Positives (FP)**.
- Las rutas que aparecen solo en el JSON esperado son **False Negatives (FN)**.

[1] https://huggingface.co/Qwen/Qwen3-0.6B-Base

Cálculo del puntaje

La métrica usa una versión ponderada del F1 score, que da más importancia a los valores correctos que a las llaves bien estructuradas.

$$F_1 = \frac{2 \times TP}{2 \times TP + FP + FN}$$

En las ponderaciones, las coincidencias de estructura (*keys*) valen menos y las coincidencias de valores (*fields*) valen más, según los siguientes pesos.

- `key_weight = 1`
- `field_weight = 9`

Para tener más información sobre la implementación de la evaluación, por favor entrar a este enlace

4 Descripción de los Datos

train/

Esta carpeta contiene los archivos con los que deberán entrenar su modelo. Cada archivo dentro de la carpeta contiene múltiples registros, cada uno de los cuales tiene los siguientes valores:

- `natural_language`: El texto en lenguaje natural con el que se pretende crear la orden de compra.
- `json_data`: El diccionario de datos JSON que representa la orden de compra esperada.

test.json

Este es el archivo a partir del cual deberán generar su archivo de respuesta descrito en la sección de evaluación. Este contiene:

- `id`: El identificador del registro de evaluación. Deben asegurarse de que este sea el ID que quede en su archivo de entrega.
- `natural_language`: El texto del mensaje con el que se pretende crear la orden de compra.

4.1 Archivo de respuesta

Por cada `id` en el conjunto de prueba (`test.csv`), se debe incluir una fila en el archivo de envío (`submission.csv`) con una columna llamada `prediction`. En esta columna se debe colocar el **JSON generado por el modelo** como una cadena de texto (no como un objeto).

El formato del archivo debe ser:

```
id,prediction
1,"{""buyer"": {""name"": ""Laura Mueller"", ""email"": null}, ""purchases"": [...
2,"{""buyer"": {""name"": ""Juan Perez"", ""email"": null}, ""purchases"": []}"
3,"{""buyer"": {""name"": ""Pedro Sanchez"", ""email"": ""pedro@example.com""},...
...
```

Importante:

- El archivo debe estar en formato **CSV** con las columnas `id` y `prediction`.
- La columna `prediction` debe contener un **JSON válido en formato de texto**.
- Si el JSON no es válido, la métrica asignará un **puntaje de 0** para ese registro.
- El orden de los elementos dentro de las listas (`[]`) **no afecta** la puntuación.

5 Rúbrica de la nota del proyecto

La nota del proyecto de la semana 8 se compone de la forma descrita en la siguiente tabla:

Table 2: Composición de la nota del proyecto de la semana 8

Item	**Porcentaje de la nota**
Participación en la competencia	20%
Superar la linea base	40%
Percentil obtenido en la competencia	40%

5.1 Participación en la competencia

Para obtener el 20% de la nota, usted y su grupo deberán realizar por lo menos 5 envíos diferentes a la competencia de Kaggle.

5.2 Superar la linea base

El 40% de la nota corresponderá a la capacidad de su modelo para superar el desempeño del modelo base en la tarea evaluada. Es decir, si su modelo obtiene un rendimiento menor o igual al del modelo base, su equipo recibirá una calificación de 0.0 en este criterio. En caso contrario, obtendrán una calificación de 5.0.

El desempeño del modelo base puede consultarse en la tabla de líderes de Kaggle, bajo el nombre `benchmark.csv`. No obstante, debe tenerse en cuenta que este desempeño puede variar respecto al que se observe al cierre de la competencia, ya que se reserva un subconjunto del conjunto de evaluación para la evaluación final. Por esta razón, para la aplicación de este criterio, se considerará que el promedio de desempeño de su modelo en los dos tableros (público y privado) es superior al promedio del modelo base.

5.3 Percentil obtenido en la competencia

El 40% restante de la nota final se obtiene de acuerdo con el desempeño del modelo desarrollado con su grupo en la competencia de Kaggle. la posición final a ser tenida en cuenta para su nota sera su posición en el leader-board privado. La nota sera asignada de acuerdo a la siguiente tabla

Desempeño obtenido	**Porcentaje del rubro**
$<$ percentil 25	0%
$\geq$ percentil 25 y $<$ percentil 50	50%
$\geq$ percentil 50 y $<$ percentil 75	75%
$\geq$ percentil 75	100%

Table 3: Distribución del porcentaje del rubro según desempeño obtenido

6 Entregables, Reglas y Restricciones

Recuerde que su sistema no tendrá validez si no cumple los requisitos que se mencionan a continuación. Por lo tanto, obtendrá una calificación de 0 en la actividad correspondiente al proyecto final.

6.1 Estructura de Entregables

Deberán entregar los siguientes archivos con los nombres descritos

- `train.ipynb`: En este archivo deberá incluir todos los pasos necesarios para el entrenamiento de su modelo correctamente organizado y documentado.
- `weights.pt`: Son los pesos resultantes del entrenamiento. Estos deben ser entregados para que los resultados de su modelo sean replicados por el equipo docente.

- `evaluation.ipynb`: Este archivo debe incluir las funciones creadas para la generación de los JSONs para los correos de prueba de la partición de test. Asegúrense de que este archivo cargue los pesos de `weights.pt` y lea directamente el archivo test.json.
- `open-evaluation.ipynb`: este *notebook* debe contener las funciones necesarias para realizar una consulta abierta en lenguaje natural y obtener una respuesta generada por el modelo.

Recuerde seguir las normas de entrega de *notebooks* para los micro-proyectos establecidas, y asegurarse de que todos los archivos estén completamente ejecutados antes de la entrega.

6.2 Entorno de Ejecución

El equipo del curso ejecutará los notebooks en Google Colab o en el entorno de GPUs de Coursera. Si se requieren configuraciones específicas, incluir un archivo `instrucciones.txt` con los pasos necesarios, así como el archivo `requirements.txt` de su ambiente de python.

6.3 Reglas

Para esta competencia:

- Como modelo base solo se permite el uso de 'Qwen/Qwen3-0.6B-Base'.
- No se permite el uso de ningún otro modelo preentrenado, ni que se consuma vía API.
- No se permite el uso de datos de testing en el entrenamiento de ninguna manera.
- No se permiten datos adicionales de entrenamiento, en particular generaciones sintéticas.

7 Fechas Importantes

- Fecha de finalización: **30/11/2025 11:55 PM**

8 Envío de Resultados

Para enviar las respuestas a Kaggle, se debe generar un archivo CSV en el formato indicado y subirlo en la plataforma. Una parte de las respuestas se usará para generar un **score público**, mientras que otra parte se reserva para el **score privado**, que determinará el resultado final.

9 Enlaces de la Competencia

- **Enlace de Invitación**: Kaggle Invitation
- **URL de la Competencia**: Competencia MAIA PLN 2025