# Desarrollando un chatbot para refugiados: nuestra experiencia en Saturdays.AI

## Contexto y equipo

Hoy en día tenemos a disposición teléfonos móviles, desde los accedemos a todo tipo de información y aplicaciones. Aun así, hay tareas que se nos siguen resistiendo... Como sería el caso de encontrar la información sobre un trámite en específico. Algunas veces acabamos perdiéndonos en los menús y subpáginas de las páginas oficiales… hasta que llegamos al trámite que buscamos. ¡Eureka!

(Bueno sí, “eureka” por el momento, porque otra cosa es que nos falte algún dato por aportar... y que cuando consigamos la documentación, seamos capaces de repetir la búsqueda sin tener que invertir casi el mismo tiempo que la primera vez…!)

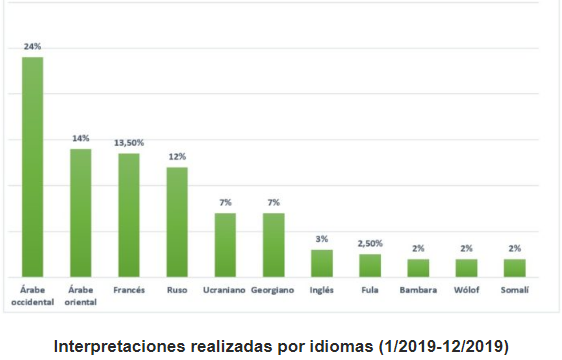
Muy bien, esto sucede a aquellos que entienden el idioma e incluso han visitado las páginas oficiales en varias ocasiones, para uno u otro tipo de trámites. ¿Nos podemos imaginar cómo sería si jamás hubiésemos visitado estas páginas? ¿Y si además, la información no estuviese en nuestro idioma nativo? Por no hablar de estar tratando de localizar información para regularizar nuestra situación en el país destino, sin poder recurrir al país origen a por orientación…

A continuación compartimos nuestro planteamiento: Supongamos que debes salir de tu país en el que sufres por tu integridad. Llegas a España, y consideras el solicitar protección internacional. Ni siquiera hablas el idioma del país de destino… ¿Dónde puedes dirigirte? ¿Cuáles son los primeros trámites que te van permitir regularizar tu situación?

Consigues llegar a un servicio de atención… tienes muchas preguntas. Ves que están colapsados y no solo con eso, además tienes la dificultad del idioma.

Como habréis imaginado, en este caso nos enfocamos en el grupo de refugiados; actualmente los servicios se saturan, llegándose a posponer la atención en algunos casos. En esta situación, ¿sería útil disponer de un aplicativo que pueda responder a las primeras consultas o por lo menos las más generales y en base a las últimas actualizaciones, de manera que los servicios pudieran enfocarse en las especificidades de cada caso?

Decidimos ponernos las manos a la obra preparando un chat de atención para este grupo específico, que pudiera proveer respuestas generales entorno a esta situación y atender en diferentes idiomas.



Fuente: <https://www.cear.es/projects/traduccion-e-interpretacion/>

Antes de ponernos de lleno en los detalles, nos presentamos:

Este equipo de trabajo se conformó dentro Saturdays AI:

“[Saturdays AI](https://medium.com/saturdays-ai) es una organización con la misión de formar a las personas para que aprendan Inteligencia Artificial de forma colaborativa, y basada en proyectos de impacto social.”

Siendo un equipo interdisciplinar en ciencias de la salud, sociales y desarrollo de aplicaciones, nos disponemos a preparar un chatbot para resolución de las primeras consultas que pueda tener una persona que llega a España y en la situación de refugiado.

Nuestro objetivo es brindar información específica y relevante a las consultas de los refugiados, para facilitarles el acceso a recursos y servicios que les sean útiles.

Sobre nosotros…

[WeiXin Ivy Chen](https://www.linkedin.com/in/weixinivy/)

[Antonio Linde Medina](https://www.linkedin.com/in/antoniolinde/)

[Josue Huaman](https://www.linkedin.com/in/josue-huaman-meza/)

[Silvia Miro Sarradell](https://www.linkedin.com/in/silvia-miro-sarradell/)

El proyecto tal y como ha transcurrido, puede dividirse en dos fases:

## Fase 1

### Método 1

Nos centramos en desarrollar un chatbot al que partiendo de una pregunta que le planteemos, obtengamos una respuesta de un contexto (páginas web).

Para ello estuvimos revisando algunas opciones en [Rasa](https://rasa.com/), [HuggingFace](https://huggingface.co/) o [Dialogflow](https://cloud.google.com/dialogflow?hl=es).

El funcionamiento de un bot elaborado con Rasa o Dialogflow es el siguiente:

* El usuario introduce un texto.
* El bot clasifica el texto asociándolo a alguna de las intenciones que éste reconoce.
* Una vez que se sabe cuál es la intención del usuario, se ejecuta una acción.
* Si fuese necesario, el bot pedirá información adicional al usuario.

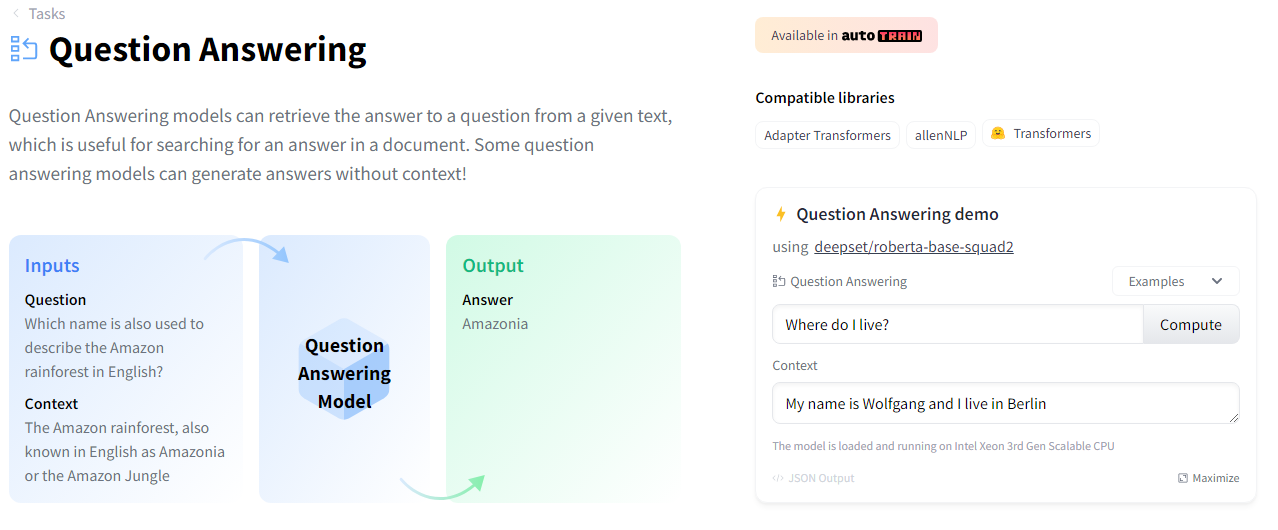
El problema que nos encontramos con este enfoque es que debemos tener definidas previamente tanto las intenciones que reconoce el bot como las acciones que ejecutará. Lo que nos interesaba era explorar opciones para realizar esto de manera automática: Que el bot aprendiera a partir de texto relativo a un tema. Y eso es que hemos intentado en este proyecto.

Todas estas opciones parecían buenas y atractivas, no obstante decidimos buscar el dinamismo en el diálogo y el poder adaptar los pesos de los modelos, por lo que finalmente decidimos explorar opciones en HuggingFace.

\*\*\*alinde! Ayuda en incluir diferencias Rasa-Dialogflow y por qué nos decantamos por HuggingFace. Gracias!!

Al indagar, vimos que en esta misma plataforma consta bastante documentación explicativa y una página de búsqueda de modelos preentrenados, que pueden usarse directamente o haciendo fine-tunning de los mismos. En este caso nos decantamos por modelos de [Question Answering](https://huggingface.co/tasks/question-answering). Probaríamos diferentes modelos de este tipo y veríamos cuál se ajustaba más a nuestro objetivo.

Como opciones de cara a la creación de una webapp, consideramos la subida del modelo final a HuggingFace como una posible opción, dado que en la página de información de los modelos publicados se facilita la documentación relacionada, contexto y API para probar el funcionamiento del modelo (\*\*\*Veis bien el vocabulario? Tenemos enlace finalmente sobre este paso?). De este modo podíamos contribuir a la comunidad con nuestro modelo ajustado y a su vez aprovechar la caja habilitada en la misma plataforma para el testeo de funcionamiento de cada modelo:



### Dataset1

Habíamos localizado el tipo de modelo a entrenar; y debíamos prepararnos para hacer el fine-tunning del modelo. Debíamos entender qué formato debían tener los datos para poder incluirlos en el fine-tunning.

Partimos de artículos del BOE para armar nuestro dataset;

Sobre estos artículos, nos disponíamos a formular preguntas y subrayar las respuestas dentro de cada artículo.

Descubrimos que existe una plataforma, [Haystack](https://annotate.deepset.ai/) que permite preparar el dataset interactuando a nivel usuario; 1)Importando los contextos, 2)incluyendo las preguntas y 3)subrayando los contextos dentro de la misma plataforma. Tras ello, la misma plataforma ofrece la opción de descargar los datos en formato csv, excel o JSON (en este último caso, se obtiene un documento JSON con el formato equivalente al formato de datos para los modelos BERT).

### Resultado1

Llegamos a incluir los datos procedentes del BOE en diferentes modelos BERT y similares, y la mayoría de modelos funcionaron proveyendo respuestas consistentes a las preguntas formuladas. Cabe considerar que el modelo con mejor resultado y consistencia en respuestas fue el modelo [RoBERTa](https://huggingface.co/docs/transformers/model_doc/roberta), recogido en el notebook “04\_12\_ES\_Roberta\_URL\_Datactalizado\_SimpleTransformersQA\_Copia\_2022\_12\_06”, disponible en github.

### Consideraciones1

Ahora nos quedaba valorar si ampliar el dataset, subir el modelo a HuggingFace y validar que la caja para testeo funcionase; o incluso ver si añadir nuevas funcionalidades, para que el modelo fuese capaz de responder a diferentes tipos de consulta (no solo de tipo “legal”).

No obstante... hubo algo que no nos acababa de encajar. A la hora de poner el chatbot en productivo, queríamos que se tratase de un aplicativo amigable: aunque este modelo cumple la función de localizar la información requerida dentro de un contexto, no era capaz de responder de forma cordial a un "Hola" o a un "Qué tal".

Estas consideraciones nos llevaron a la segunda fase de nuestro proyecto;

## Fase 2

### Contexto2

Consideramos diferentes opciones para solucionar la situación anterior.

* Podíamos incluir algunas preguntas-respuestas programadas, y tras ello un botón para activar la funcionalidad de uso del modelo;
* Podíamos usar modelos conversacionales, que integran ambas funcionalidades;
* Podíamos decantarnos por un modelo de tipo de generación de texto, y entrenarlo para que supiera responder de forma adecuada a consultas relativas al grupo de refugiados;
* Podíamos integrar diferentes modelos, uno para las preguntas-respuestas específicas y otro más "humano".

### Método2

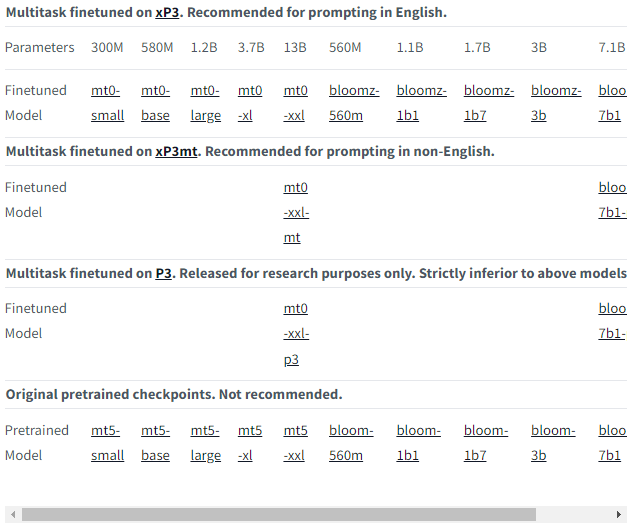
Y como os podréis imaginar, como veníamos participando de un curso de IA, decidimos decantarnos por alguna de las tres últimas opciones, considerando que la primera podría haber sido una opción en caso de haber de entregar un Producto Mínimo Viable (MVP) en cualquier otro contexto. Y de estas tres… valoramos según:

* En cuanto al uso de modelos conversacionales, al revisar en HuggingFace vimos que no constaban modelos de este tipo en idioma español; por lo que por el momento decidimos descartar esta opción;
* Junto con el auge de los modelos de generación de texto ([OpenAI](https://chat.openai.com/chat), [GPT-3](https://es.wikipedia.org/wiki/GPT-3)…), decidimos decantarnos por esta opción y probarlo en consultas relativas a nuestro objetivo; y en caso de no funcionar de forma óptima, tratar de hacer fine-tunning del modelo con nuestros datos;
* En cuanto a la integración de diferentes modelos, cada uno con un mejor performance en una u otra área, decidimos considerarlo tras haber obtenido buenos resultados en los modelos que podrían formar parte de esta solución.

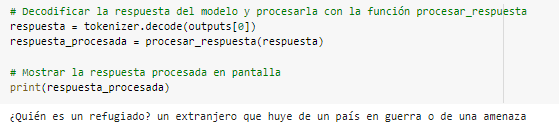
En paralelo estuvimos trabajando en la implementación del modelo resultado a [Streamlit](https://streamlit.io/), para poder proveer demo de la solución.

### Resultado2

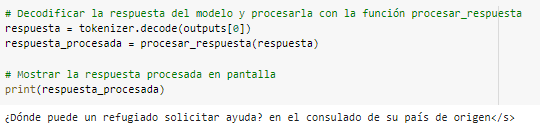
A través de HuggingFace localizamos el modelo [Bloom](https://huggingface.co/bigscience/bloom), un modelo de generación de texto. Tras probar diferentes versiones del modelo preentrenado, vimos que la versión que podíamos llegar a implementar desde google colab era la 1b1:



Y efectivamente llegamos a obtener respuestas consistentes a preguntas generales:



Al probar de realizar algunas consultas sobre el grupo de refugiados, vimos que las repuestas no se ajustaban a una resolución real. En algunos casos podían incluso llegar a suponer un riesgo. Por ejemplo:



\*\*\*Incluir disclaimer también en demo Streamlit: Este es un bot de prueba y los textos son generados. ¡No consideres esta información como fiable! Es preferible que consultes con tu especialista o asesor legal.

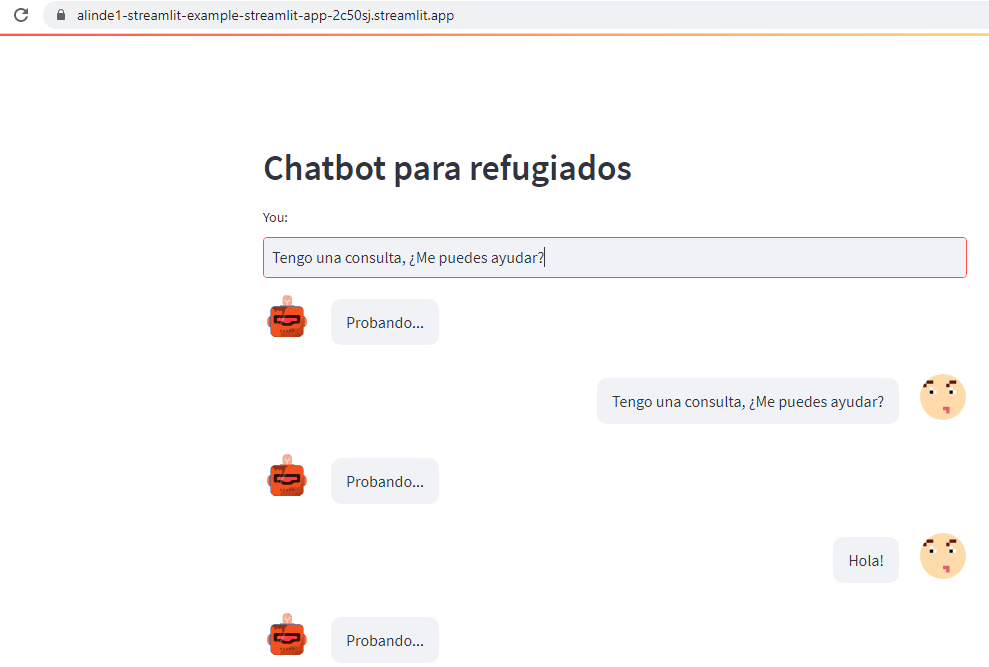
Añadido aviso en Streamlit.

Por ello al realizar este primer testeo, consideramos imprescindible realizar fine-tunning del modelo incorporando por lo menos los datos de los que disponíamos, al modelo preexistente. Os compartimos algunos detalles sobre este proceso en el siguiente apartado.

En cuanto a la demo de la solución, estuvimos preparándola en Streamlit; compartimos a continuación una demo de la solución actualizada, en el enlace siguiente. Y en el caso de cualquier cambio, prevemos informarlo en github:

\*\*\*alinde, según comentado, ayuda en este punto!! → Actualizado enlace.

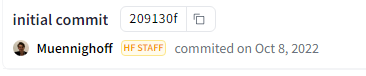
https://alinde1-streamlit-example-streamlit-app-br9pjp.streamlit.app/



### Dataset2

Esta fase del proyecto se desarrolló en Diciembre/2023;

Bloom es un modelo de publicación reciente; se subió a HuggingFace en Octubre/2022:



<https://huggingface.co/bigscience/bloomz-1b1/commits/main>

Ello nos conduce a que, al tratar de localizar información en web, nos hemos encontrado con dificultades de cara a localizar documentación explicativa y formato de los datos.

Entonces estuvimos trabajando en la adaptación del dataset para la ejecución de trainer.train() en base a la documentación disponible y más adelante, incluso partiendo de los errores recibidos al ejecutar el notebook. No obstante por el momento no hemos alcanzado a aplicar ajustes en el modelo.

### Consideraciones2

En cuanto a la adaptación de los datos para para avanzar en el fine-tunning del modelo, consultamos al chatbot de OpenAI y nos indicó las siguientes consideraciones:

Cada línea del archivo debe ser un diccionario JSON que contenga al menos dos campos: "prompt" y "response". El campo "prompt" es el texto que se usará como entrada para generar una respuesta, mientras que el campo "response" es el texto esperado como respuesta. Los datos deben estar limpios y procesados previamente para evitar problemas en el entrenamiento del modelo.

Ejemplo:

Copy code

{"prompt":"¿Qué es un refugiado?", "response":"Un refugiado es una persona que ha huido de su país debido a la persecución, el conflicto armado, la violencia generalizada o violaciones graves de los derechos humanos, y que no se puede o no quiere regresar a ese país debido al temor a represalias."}

Una vez que tengas tu conjunto de datos en el formato correcto, puedes utilizar la herramienta de fine-tuning de OpenAI para entrenar tu modelo con tus datos específicos. Es importante mencionar que para hacer un fine tuning de calidad es necesario contar con un gran volumen de datos, y que estos sean relevantes y de calidad.

Sin embargo, la información de la que parte el chat de generación de texto de OpenAI engloba hasta 2021; cuando el modelo Bloom no estaba aún disponible en HuggingFace.

No obstante, cabría valorar otras opciones en cuanto a la solución final:

## Consideraciones finales

A lo largo del proceso, nos enfrentamos a varios desafíos, como la necesidad de encontrar una forma de brindar respuestas relevantes y precisas a las consultas de los refugiados. También tuvimos que aprender a trabajar con herramientas y tecnologías nuevas. Sin embargo, estos desafíos también nos permitieron aprender mucho y desarrollar habilidades valiosas.

Consideramos que la filosofía (explícita) de HuggingFace es una práctica realmente interesante, que nos lleva a optimizar los recursos considerando también el impacto en el medio ambiente, así como conectarnos a una comunidad global. Esta plataforma nos ha sido de gran ayuda de cara a localizar modelos y comprender los avances en este ámbito.

Aun así, hemos visto que hacer el fine-tunning de los datos ¡no siempre es tan sencillo! En caso que por ejemplo tú, como lector de este documento, dispusieses de alguna información sobre este tema... ¡estaríamos encantados de leer tus comentarios!

Creemos que los próximos pasos del proyecto podrían basarse en:

1)Seguir trabajando en el fine-tunning del modelo de generación de texto,

2)Adaptar algún modelo conversacional al español, o bien

3)Implementar un enfoque mixto:

- Saludos y preguntas-respuestas generales a través de un modelo de generación de texto, y

- Preguntas-respuestas específicas a través de otro modelo, como por ejemplo el modelo de Question Answering propuesto en la Fase1 de este proyecto. Y de este modo la información siempre estaría actualizada en base a publicaciones.

- En este caso, el chat interpretaría de qué tipo de input se trata en cada momento y aplicaría uno u otro modelo.

Otros pasos podrían ser contrastar el funcionamiento en diferentes idiomas, e incluso valorar la opción de que el chat pudiese funcionar también por voz.

Aunque todavía queda trabajo por hacer, estamos emocionados de haber llegado hasta aquí en el desarrollo de nuestro chatbot para refugiados. Creemos que este proyecto tiene el potencial de marcar una gran diferencia en la vida de las personas que huyen de sus hogares y buscan una nueva oportunidad. Continuaremos trabajando en mejorar nuestro chatbot, haciendo fine tuning específico para el área de refugiados y probando el chatbot en diferentes idiomas. Esperamos poder compartir nuestros resultados y aprendizajes con la comunidad, con la esperanza de inspirar a otros a unirse a nosotros en esta misión.

Ah, y por cierto. Habréis visto que algunos de los párrafos de esta publicación se presentan en azul. Se trata de párrafos generados por el chat de OpenAI, al consultarle cómo nos proponía exponer las conclusiones de nuestro proyecto 

Gracias AI Saturdays por esta gran oportunidad!

## Dejamos a disposición...

### Sobre el proyecto:

* [Repositorio GitHub](https://github.com/josuemzx/Chatbot-para-refugiados)
* [Streamlit](https://alinde1-streamlit-example-streamlit-app-br9pjp.streamlit.app/)
  + \*\*\*alinde, ¿nos podrías ayudar incluyendo el enlace final streamlit en este punto? Gracias! Actualizado el enlace.
* [Saturdays AI](https://medium.com/saturdays-ai)

### Sobre tratamiento de datos:

* [Fuente de datos: BOE](../../../../../../C:/Users/simis/Documents/GitHub/Chatbot-para-refugiados/doc/o%09https:/www.interior.gob.es/opencms/es/servicios-al-ciudadano/tramites-y-gestiones/oficina-de-asilo-y-refugio/)
* [Haystack](https://annotate.deepset.ai/)

### Sobre modelos:

* [HuggingFace](https://huggingface.co/)
  + [Question Answering](https://huggingface.co/tasks/question-answering)
    - [RoBERTa](https://huggingface.co/docs/transformers/model_doc/roberta)
  + Generación de texto
    - [Bloom](https://huggingface.co/bigscience/bloom)
      * <https://huggingface.co/bigscience/bloomz-1b1/commits/main>
* [OpenAI](https://chat.openai.com/chat)
* [GPT-3](https://es.wikipedia.org/wiki/GPT-3)

### Otros enlaces de interés:

* [Rasa](https://rasa.com/)
* [Dialogflow](https://cloud.google.com/dialogflow?hl=es)
* <https://www.cear.es/projects/traduccion-e-interpretacion/>