Datasets

# HumanEval

Github: <https://github.com/openai/human-eval>

Paper: <https://arxiv.org/pdf/2107.03374.pdf>

Descripción: dataset de problemas de programación que incluye docstrings y signaturas, acompañado de una solución y tests unitarios.

Lenguajes: Python

Código incluido:

* Evaluación de modelos contra el dataset

Métricas de evaluación:

* Pass@k: se generan k muestras por cada problema y se considera resuelto si alguno de las k muestras pasa todos los tests del problema. Se reporta la fracción de problemas considerados resueltos.
* Pass@k modificado: se generan n >= k muestras por problema para calcular un estimador de pass@k.

Resultados:

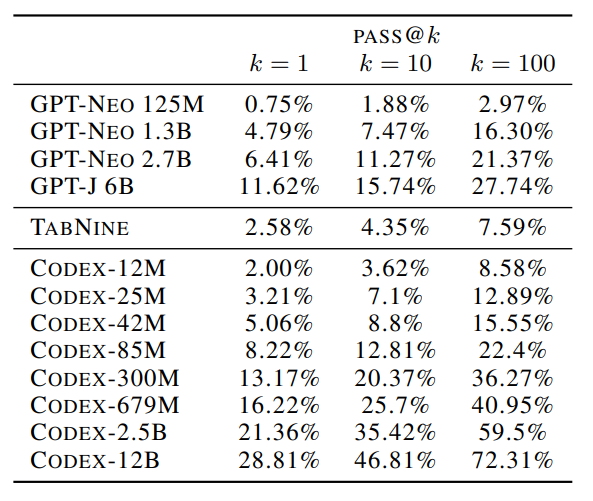


Ilustración 1. Table 1. Codex, GPT-Neo, & TabNine evaluations for HumanEval.

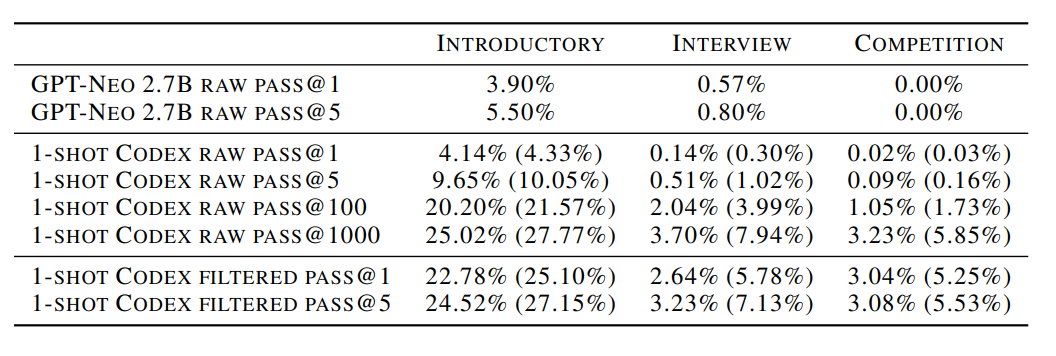


Ilustración 2. Finetuned GPT-Neo numbers from the APPS paper referenced above. For Codex-12B, the number of passing programs that timeout on some test is in the bracket.

# APPS

Github: <https://github.com/hendrycks/apps>

Paper: <https://arxiv.org/pdf/2105.09938.pdf>

Descripción: dataset de problemas de programación que incluye la definición del problema y ejemplos de entrada y salida, acompañado de soluciones correctas y tests unitarios. Los problemas también están etiquetados según el nivel de dificultad. El formato de los problemas puede estar en “Call-Based Format”, en el cual se incluye la cabecera de la función que deberá devolver el resultado, o en “Standard Input Format”, en el cual la solución deberá leer del canal estándar de entrada y sacar el resultado por el canal de salida.

Lenguajes: Python

Código incluido:

* Evaluación de modelos contra el dataset
* Entrenamiento de modelos GPT
* Pesos de modelos entrenados (GPT-2 1.5B y GPT-Neo 2.7B)

Métricas de evaluación:

* Test Case Average: media del porcentaje de test cases pasados para cada problema
* Strict Accuracy: porcentaje de problemas en que se pasan todos los test cases

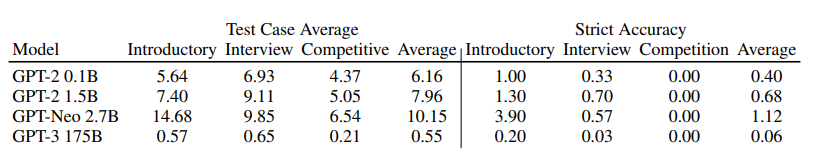
Resultados:

Ilustración 3. Table 2: Average percentage of test cases passed and strict accuracy for each model and difficulty level. All values are percentages. Note ‘0.1B’ indicates the number of model parameters in billions. GPT-3 is a few-shot model and not fine-tuned, unlike the other models.

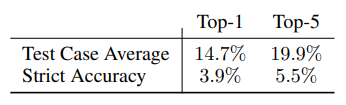


Ilustración 4. Table 3: GPT-Neo 2.7B performance on introductory problems using one generated program (Top-1) and the best of five generated programs (Top-5)

# CodeContests

Github: <https://github.com/deepmind/code_contests>

Paper: <https://storage.googleapis.com/deepmind-media/AlphaCode/competition_level_code_generation_with_alphacode.pdf>

Descripción: dataset con problemas de programación acompañados de soluciones (correctas e incorrectas) y tests unitarios (visibles e invisibles en tiempo de competición). Los problemas también están etiquetados según el nivel de dificultad y las estrategias de programación necesarias para resolverlos. Los datos están obtenidos de la plataforma CodeForces y combinados con los datasets existentes Description2Code y CodeNet. El set de validación consiste solo de nuevos problemas publicados en CodeForces.

Lenguajes: C++, Python y Java

Código incluido: -

Métricas de evaluación:

* pass@k
* 10@k: igual que pass@k pero en este caso, solo 10 de las k muestras son evaluadas.

# CodeNet

Github: <https://github.com/IBM/Project_CodeNet>

Paper: <https://arxiv.org/pdf/2105.12655.pdf>

Descripción: dataset de problemas de programación acompañados con soluciones (correctas e incorrectas). Otros metadatos presentes en el datatset (no para todos los casos) incluyen dificultad del problema, ejemplos de entrada y salida, lenguage, compilador, tamaño del código, uso de memória, tiempo de uso de CPU y estado (aceptado o no acpetado).

Lenguajes: +50. Los dominantes son: C++, C, Python y Java. Para más detalle ver Project\_CodeNet\_statistics.xlsx. (1277 submissions en COBOL, 727 aceptadas y 550 no aceptadas).

Codigo incluido:

* Analisis del dataset
* Tokenizador
* Generador de AST
* Búsqueda de códigos duplicados
* Análisis de código para la construcción de grafos de flujo de control y datos

Métricas de evaluación: -

# GPT-Code-Clippy (GPT-CC)

Github: <https://github.com/CodedotAl/gpt-code-clippy>

Descripción: dataset de archivos de código extraídos de Github combinado con el dataset The Pile. Los metadatos incluyen el nombre del repositorio, el número de estrellas de este y el lenguaje, entre otros. Existe una versión de este dataset con las duplicaciones eliminadas.

Lenguajes: varios. Principalmente JavaScript y Python.

Código incluido:

* Entrenamiento de modelos sobre GPT-2 y GPT-Neo
* Modelos entrenados sobre el propio dataset, APPS y Code Search Net Challenge.

Métricas de evaluación:

* pass@k

Resultados:

# CodeXGLUE

Github: <https://github.com/microsoft/CodeXGLUE>

Paper: <https://arxiv.org/pdf/2102.04664.pdf>

Descripción: Múltiples datasets i modelos para diferentes tareas: code-code, text-code, code-text, text-text. Para la tarea de text-code se usa el dataset Concode (Java), orientado a generar código teniendo en cuenta el contexto que rodea al método (clase de Java). Los atributos de la clase de las que se han extraído los métodos de ejemplo forman parte del input, acompañando a la descripción de la funcionalidad del código. CodeGPT-adapted tiene una performance buena en Code-Text generation.

Lenguajes: Java

Código incluido:

* Entrenamiento de modelos GPT (y otros)
* Modelos entrenados
* Evaluación

Métricas de evaluación:

* EM
* BLUE
* CodeBLEU

# Code Search Net

Github: <https://github.com/github/CodeSearchNet>

Paper: <https://arxiv.org/pdf/1909.09436.pdf>

Descripción: colección de datasets y benchmarks para la tarea de buscar código apropiado (dentro del propio dataset) dado un texto descriptivo de una funcionalidad: datos, modelos de referencia con pesos pre-entrenados, métricas de referencia y herramientas de evaluación. Los datos están compuestos de pares (texto, código) dónde el texto es un comentario de alto nivel (docstring) de lo que implementa el código (función).

Lenguajes: Go, Java, JavaScript, PHP, Python y Ruby

Código incluido:

* Descarga de datos
* Pre-procesamiento
* Entrenamiento de modelos
* Evaluación

Métricas: Normalized Discounted Cumulative Gain.

# PyTorrent

Github: <https://github.com/fla-sil/PyTorrent>

Paper: https://arxiv.org/pdf/2110.01710.pdf

Descripción: dataset de código Python acompañado de su respectivo docstring en el código.

Lenguajes: Python

Código incluido:

* Modelo pre-entrenado RoBERTa-MLM-based

Métricas de evaluación: -

# Soluciones

* Codex
* AlphaCode
* GPT-Neo (APPS)
* CodeGPT (CodeXGLUE)
* GPT-CC (CodeClippy)
* GPT-2, GPT-3 (¿?)

