Procesamiento del Lenguaje Natural, Revista nº 73, septiembre de 2024, pp. 221-233 recibido 03-04-2024 revisado 03-05-2024 aceptado 15-05-2024

# Balancing Efficiency and Performance in NLP: A Cross-Comparison of Shallow Machine Learning and Large Language Models via AutoML

## *Equilibrando eficiencia y rendimiento en PLN: comparación cruzada de Machine Learning Tradicional y Grandes Modelos de Lenguaje mediante AutoML*

**Ernesto L. Estevanell-Valladares[1,2], Yoan Gutiérrez[2], Andrés Montoyo-Guijarro[2], Rafael Muñoz-Guillena[2], Yudivián Almeida-Cruz[1]**
[1]Universidad de la Habana
[2]Universidad de Alicante
elev1@alu.ua.es

**Abstract:** This study critically examines the resource efficiency and performance of Shallow Machine Learning (SML) methods versus Large Language Models (LLMs) in text classification tasks by exploring the balance between accuracy and environmental sustainability. We introduce a novel optimization strategy that prioritizes computational efficiency and ecological impact alongside traditional performance metrics leveraging Automated Machine Learning (AutoML). Our analysis reveals that while the pipelines we developed did not surpass state-of-the-art (SOTA) models regarding raw performance, they offer a significantly reduced carbon footprint. We discovered SML optimal pipelines with competitive performance and up to 70 times less carbon emissions than hybrid or fully LLM pipelines, such as standard BERT and DistilBERT variants. Similarly, we obtain hybrid pipelines (using SML and LLMs) with between 20% and 50% reduced carbon emissions compared to fine-tuned alternatives and only a marginal decrease in performance. This research challenges the prevailing reliance on computationally intensive LLMs for NLP tasks and underscores the untapped potential of AutoML in sculpting the next wave of environmentally conscious AI models.
**Keywords:** Natural Language Processing, Machine Learning, AutoML, LLM.

**Resumen:** Este estudio analiza críticamente la eficiencia de recursos y el rendimiento de los métodos de Aprendizaje Automático Superficial (SML) frente a los Grandes Modelos de Lenguaje (LLM) en tareas de clasificación de texto explorando el equilibrio entre precisión y sostenibilidad medioambiental. Se introduce una novedosa estrategia de optimización que prioriza la eficiencia computacional y el impacto ecológico junto con las métricas de rendimiento tradicionales aprovechando el Aprendizaje Automático de Máquinas (AutoML). El análisis revela que, si bien los pipelines desarrollados no superan a los modelos SOTA más avanzados en cuanto a rendimiento bruto, reducen significativamente la huella de carbono. Se descubrieron pipelines óptimos de SML con un rendimiento competitivo y hasta 70 veces menos emisiones de carbono que pipelines híbridos o totalmente LLM, como las variantes estándar de BERT y DistilBERT. Del mismo modo, obtenemos pipelines híbridos (que incorporan SML y LLM) con entre un 20% y un 50% menos de emisiones de carbono en comparación con las alternativas fine-tuneadas y sólo una disminución marginal del rendimiento. Esta investigación pone en cuestión la dependencia predominante de los LLM de alta carga computacional para tareas de PLN y subraya el potencial sin explotar de AutoML para esculpir la próxima oleada de modelos de IA con conciencia medioambiental.
**Palabras clave:** Procesamiento del Lenguaje Natural, Aprendizaje Automático, AutoML, LLM.

ISSN 1135-5948 DOI 10.26342/2024-73-16 

## 1 Introduction

In the rapidly evolving domain of Natural Language Processing (NLP), the advent of Large Language Models (LLMs) such as BERT (Devlin et al., 2018), GPT-3 (Brown et al., 2020) and GPT-4 (OpenAI, 2023) has brought significant advancements, enhancing model capabilities across a myriad of tasks. Despite their impressive performance, the environmental impact of training and deploying these models has become a growing concern, highlighting the need for sustainable AI practices (Faiz et al., 2023; Dodge et al., 2022). Concurrently, the efficient and task-specific nature of Shallow Machine Learning (SML) techniques (Kowsari et al., 2019) suggests a potential pathway to achieving high performance in NLP tasks while mitigating environmental costs. However, these techniques also face challenges related to their task-specific nature and the complexity involved in selecting optimal models and parameter configurations (Hutter, Kotthoff, and Vanschoren, 2019).

This study addresses the ecological challenges associated with the significant computational demands of deep learning models (Schwartz et al., 2019) in NLP model development. Our methodology leverages Automated Machine Learning (AutoML) (Hutter, Kotthoff, and Vanschoren, 2019; Thornton et al., 2013) to create and optimize LLM-SML hybrid pipelines, incorporating multi-objective optimization of performance and resource consumption metrics for Text Classification. This approach aims to harness the pre-trained language representations of LLMs (Qiu et al., 2020), exploring their potential to achieve satisfactory results in different domains when paired with SML techniques without fine-tuning, thus significantly reducing their environmental footprint.

We aim to develop and evaluate SML-only and LLM-SML hybrid pipelines for text classification, emphasizing sustainable AI by simultaneously optimizing performance and resource-consumption metrics. By integrating performance and environmental impact assessments into our evaluation, we seek to demonstrate how these models can achieve competitive task-specific accuracy with minimal resource consumption.

The remainder of the paper is organized as follows: Section 2 reviews related work, highlighting efforts to measure and mitigate the environmental impact of ML models. Section 3 describes our methodology, including the selection criteria for AutoML systems and our approach to integrating SML and LLM models. Section 4 details the experimental setup and presents our findings, followed by a discussion in Section 5 that interprets these results within the broader context of sustainable AI. Finally, Section 6 concludes the paper with reflections on the implications of our study and suggestions for future research directions.

## 2 Related Work

This section summarizes the main advances in effectively measuring, reporting, and mitigating the carbon emissions generated by modern ML techniques and optimizing resource consumption. Equally important, we review attempts to compare SML methods with LLMs in terms of performance and resource consumption efficiency.

Numerous studies assess the environmental impact of LLM solutions (Thompson et al., 2021; Wang et al., 2023), which are becoming increasingly popular and require significant resources. The performance of these models typically scales with the model size, dataset size, and the amount of computing used for training (Kaplan et al., 2020). Several studies have consistently reported alarming numbers on the carbon footprint of LLMs as the models grow larger (Anthony, Kanding, and Selvan, 2020; Bannour et al., 2021; Dodge et al., 2022; Faiz et al., 2023).

Efforts to address the scalability issue of LLMs include using sparsely activated Mixture of Experts (MoE) models that maintain a constant computational cost while scaling the number of parameters (Lepikhin et al., 2020; Fedus, Zoph, and Shazeer, 2022). These models have outperformed dense models in the speed-accuracy Pareto curve (Fedus, Zoph, and Shazeer, 2022).

Another promising field for creating compute-efficient LLMs is Knowledge Distillation (KD). For instance, DistilBert (Sanh et al., 2020) is a prime example of a model that retains 97% of Bert's (Devlin et al., 2018) language understanding capabilities while being 40% smaller and 60% faster (Sanh et al., 2020). Gu et al. (2023) proposes a new method scalable to larger language models, which outperforms previous KD methods.

Language models require significantly more resources for training, tuning, and inference than traditional shallow ML models. How-

ever, once trained, depending on the amount of data available and their pre-training, LLMs can be employed for many different tasks. For instance, GPT-4(OpenAI, 2023) can be used for medical tasks where it was not previously trained (Nori et al., 2023). Additionally, GPT-3 (Floridi and Chiriatti, 2020) can produce state-of-the-art results in text classification without additional fine-tuning, using a prompting strategy developed by Sun et al. (2023).

Since LLMs create universal language representations (Qiu et al., 2020), they have better generalization capabilities than SML methods. On the other hand, SML requires custom training for each downstream task, making the development of task-specific ML pipelines complicated. These challenges can nonetheless be overcome by automating the creation of task-specific ML pipelines using AutoML (Kotthoff et al., 2019).

There is a lack of comprehensive evaluation of AutoML's effectiveness and computing efficiency, especially in generating SML and LLM-SML hybrid pipelines compared to purely LLM approaches. A study by González-Carvajal and Garrido-Merchán (2020) compared the performance of BERT against SML techniques, some of which were generated by H2OAutoML (LeDell and Poirier, 2020). Their results highlighted the generalization capabilities of LLMs over SML methods. However, their study only evaluated accuracy using Term Frequency - Inverse Document Frequency (TF-IDF) (Yun-tao, Ling, and Yongcheng, 2005) for preprocessing, which limits the scope of the evaluation.

AutoML shows potential for bridging the performance gap between SML and LLM approaches while minimizing resource consumption. Therefore, we examine several AutoML systems, assessing their capability to automate and optimize SML, LLM, and LLM-SML hybrid pipeline configurations.

## 2.1 AutoML

AutoML systems are designed to automate the selection and optimization of machine learning pipelines. However, there are many solutions available that differ in their features. In particular, we consider the following features key to our research:

**Supporting multiple libraries** guarantees a rich search space of SML methods to be explored and a significant number of combinations to be evaluated.

**Including pre-trained models** allows for automatically comparing SML, LLM-SML, and LLM pipelines.

**Multi-objective optimization** can be potentially exploited for exploring the tradeoffs between resource efficiency and performance.

**Resource constrains** further allows for generating resource-efficient solutions.

Table 1 compares several existing AutoML systems, focusing on their flexibility based on the features that we have identified as relevant.

The richness of their search space is one of the strongest factors influencing the potential of AutoML systems. Including multiple libraries in the AutoML systems algorithm pools can be crucial in unlocking previously unexplored, well-performing pipelines combining techniques from multiple domains. Auto-Sklearn (Feurer et al., 2020) relies on Scikit-learn (Pedregosa et al., 2011), Auto-Weka (Kotthoff et al., 2019) on Weka (Holmes, Donkin, and Witten, 1994), and Auto-Keras (Jin, Song, and Hu, 2019) on Keras (Chollet, 2018), which restricts their use to supervised learning problems. In contrast, TPOT-NN (Romano et al., 2021), ML-Plan (Mohr, Wever, and Hüllermeier, 2018), HML-Opt (Estévez-Velarde et al., 2021), and AutoGOAL (Estevez-Velarde et al., 2019) integrate technologies from different learning libraries. Moreover, only HML-Opt and AutoGOAL can seamlessly be used for tasks from domains such as NLP.

Auto-Keras and ZAP (Öztürk et al., 2022) focus on providing deep learning-based solutions. In contrast, the other compared systems mainly focus on building SML pipelines—however, only ZAP and AutoGOAL support pre-trained models. While the latter includes both LLMs and SML techniques in its algorithm pool, Zap focuses on vision models and, hence, cannot generate NLP solutions.

Due to our requirement to balance performance and resource efficiency, selecting a system that can optimize multiple objectives simultaneously is vital. TPOT-NN implements multiobjective optimization to maximize classification accuracy and minimize

| Features \ Systems | Auto-Weka 2.0 | RECIPE | ML-Plan | Auto-Keras | Auto-Sklearn 2.0 | TPOT-NN | HML-Opt | AutoGOAL | ZAP |
|---|---|---|---|---|---|---|---|---|---|
| Support multiple libraries | | ≈ | ✓ | | | ✓ | ✓ | ✓ | |
| Includes pre-trained models | | | | | | | | ✓ | ✓ |
| Multi-objective optimization | | ≈ | | | | ✓ | | ✓ | |
| Resource constraints | | | ✓ | ✓ | ✓ | | | ✓ | |
| Year | 2017 | 2017 | 2018 | 2018 | 2019 | 2020 | 2020 | 2020 | 2022 |

Table 1: Comparison of several existing AutoML systems' capabilities to deal with heterogeneous scenarios. Entries marked with ≈ indicate that the system design theoretically supports the capability, but we have no record of its implementation.

pipeline complexity simultaneously (Olson and Moore, 2019). However, it does not allow users to set the optimization objectives. Conversely, AutoGOAL's latest version[1] allows for multiobjective optimization of any set of objectives the user can provide. Unfortunately, no other studied systems have implemented such a feature.

## *3 Methodology*

Our research targets the generation and evaluation of SML and LLM-SML hybrid pipelines that emphasize sustainable AI. We hypothesize that combining these techniques to balance performance and resource efficiency can reduce AI models' environmental impact without sacrificing performance.

We are using AutoML to automatically generate, evaluate, and compare SML and LLM-SML hybrid pipelines based on their performance and computing time to achieve this. Figure 1 shows examples of hypothetical pipelines that combine SML and LLMs in different ways to produce varying results. We aim to identify the Pareto front, representing the optimal balance between performance and resource efficiency. By doing so, we expect to discover new combinations of SML techniques and pretrained LLMs that do not require expensive fine-tuning to produce good results.

Based on our study of the current state of AutoML (see Section 2), we selected AutoGOAL as our evaluation framework. This AutoML system has a vast collection of algorithms from multiple renowned machine-learning libraries. Additionally, AutoGOAL comes with adjustable resource consumption limits whereby users can set rules for pipeline evaluation duration and maximum memory usage.

### 3.1 AutoGOAL

AutoGOAL (Estevez-Velarde et al., 2020) allows for optimizing multiple objectives simultaneously. We selected *macro* $F_1$ and *evaluationtime* for optimization due to several factors:

- Optimizing for the *macro* $F_1$ score ensures the robustness and versatility of the generated solutions, balancing precision and recall.
- Total training time has been strongly predictive of total energy consumed (kWh) (Wang et al., 2023) (under the same hardware conditions) and won't add any overhead to the models as directly measuring hardware stats on regular intervals.

#### Algorithms

AutoGOAL imports algorithms from scikit-learn (Pedregosa et al., 2011), nltk (Loper and Bird, 2002), Spacy (Honnibal et al., 2020), gensim (Řehůřek and Sojka, 2010), transformers from Huggingface, and others. A total of 128 SML algorithms from such libraries are included in our search space, of which 17 are classifiers. Figure 1 exemplifies how AutoGOAL could potentially build hybrid pipelines by utilizing LLMs as feature extractors that can be connected to any classifier.

[1]The latest version of this system with all the features is available at https://github.com/autogoal/autogoal/tree/adding-huggingface-transformers

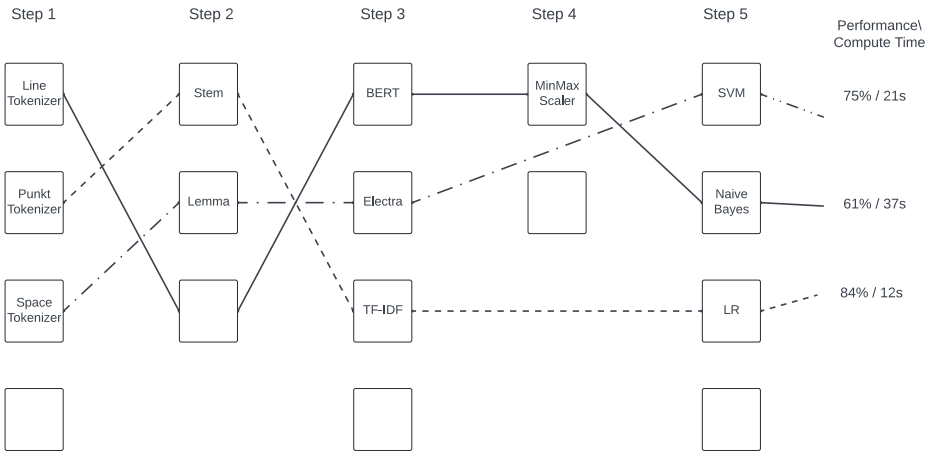



Figure 1: Illustrative example of an abstract pipeline, in which different steps can be performed to achieve varying performance levels and compute time.

A comprehensive list of all algorithms is available at the system's GitHub repository[2].

Table 2 summarizes all the LLMs in AutoGOAL's algorithm library. At the time of writing, AutoGOAL only supports LLMs for inference and has not enabled automatic fine-tuning of the models. The system allows them to be used as embedders at multiple levels by pooling lower-level embeddings (e.g., sentence embeddings can be formed by pooling word embeddings). This means all LLMs can participate in LLM-SML hybrid pipelines as AutoGOAL can select them for feature extraction, eventually attaching them to a classifier.

### 3.2 Measuring Energy and Carbon

We captured essential resource consumption statistics to estimate the carbon emissions associated with various computational pipelines using the CodeCarbon (Schmidt et al., 2021) Python library to measure carbon emissions and power consumption stats from our pipeline training and testing. Additionally, we extract CPU-usage statistics via the psutil Python library (Rodola, 2020). CodeCarbon uses pyNVML to generate GPU-related statistics for Nvidia's GPUs.

As noted by Wang et al. (2023), Recent work has found that the existing libraries and code for estimating the carbon emissions of NLP techniques vary in accuracy and generalizability to different types of hardware. They computed an expected power loss constant (close to 1) with which they calibrated CodeCarbon's readings. However, as we aim to make a comparative side-by-side analysis, we don't use such calibration as it won't affect the comparison and would require an additional infrastructure energy analysis.

## 4 *Experiments*

We executed our proposal on three different text classification tasks with varying difficulty levels: IMDB Movie Reviews (IMDB) (Maas et al., 2011), AG News (AG) (Zhang, Zhao, and LeCun, 2015), and Yelp Reviews Full (YR) (Zhang, Zhao, and LeCun, 2015). Table 3 illustrates the variation in the number of examples and the number of classes to predict. All classes are balanced among the datasets.

We ran the AutoML process for each task in two different hardware setups. Table 4 presents setups A and B. SML pipelines were generated on setup A. At the same time, we employed setup B to find LLM-SML hybrid pipelines, as it had an available GPU.

The AutoML system configuration for each experiment is displayed in Table 5. All experiments were carried out until the Global Timeout (G. To) or 10000 pipeline evaluations were reached, whichever came first. The cross-validation steps (CV) were set to divide all training data into 70/30 train/validation splits. AutoGOAL was set to optimize *macro* $F1$ and *evaluation time*.

After completing the optimization phase, we evaluated the pipelines in the Pareto frontier on corresponding test sets. This evaluation occurred solely on setup B to get fair and comparable resource consumption stats. These stats were then used to compute each

[2]https://github.com/autogoal/autogoal/tree/adding-huggingface-transformers

| LLM | Variants |
|---|---|
| BERT (Devlin et al., 2018) | (cased, uncased) base, large, base-multilingual (only cased) |
| DistilBERT (Sanh et al., 2020) | base (cased, uncased), base-multilingual (cased) |
| RoBERTa (Liu et al., 2019) | base, large |
| XLM-RoBERTa (Conneau et al., 2020) | base, large |
| DeBERTa (He et al., 2021) | base |
| DeBERTaV3 (He, Gao, and Chen, 2023) | base |
| MDeBERTaV3 (He, Gao, and Chen, 2023) | base |
| ALBERT-v1 (Lan et al., 2019) | base, large, xlarge, xxlarge |
| ELECTRA (Clark et al., 2020) | (discriminator) small, base, large |
| T5 (Raffel et al., 2020) | small, base, large, 3B, 11B |
| FLAN-T5 (Chung et al., 2022) | base, large, xxl, xl |
| GPT-2 (Radford et al., 2019) | base, medium, large, xl |

Table 2: LLMs available in AutoGOAL's algorithm pool.

| Dataset | Train S. | Test S. | Classes |
|---|---|---|---|
| IMDB | 25k | 25k | 2 |
| AG | 120k | 7.6k | 4 |
| YR | 650k | 50k | 5 |

Table 3: Dataset statistics for IMDB Movie Reviews, AG News, and Yelp Reviews Full.

| Setup | CPU | RAM | GPU |
|---|---|---|---|
| A | i9-9900K (16c) | 127 GB | None |
| B | EPYC 7742 (64c) | 1TB | A100 |

Table 4: System configurations for experiments.

| Exp. | E. To | G. To | RAM | CV |
|---|---|---|---|---|
| IMDB (A) | 1h | 65h | 30GB | 5 |
| IMDB (B) | 1h | 65h | 30GB | 3 |
| IMDB (A) | 15m | 24h | 30GB | 5 |
| AG (B) | 1h | 48h | 30GB | 3 |
| YR (A) | 20m | 24h | 35GB | 5 |
| YR (B) | 1h | 48h | 30GB | 3 |

Table 5: AutoGOAL's configuration for each experiment (experiment's dataset and system setup, pipeline evaluation timeout, global search timeout, RAM limit in evaluation, and amount of stratified cross-validation steps).

pipeline's carbon dioxide emissions during training and testing.

### 4.1 Results

We discovered a large number of valid pipelines for all tasks. Table 6 summarizes all the pipeline evaluations conducted in the AutoML search and optimization procedure using the preset configurations (See Table 5). The columns tagged with "A" refer to runs on setup A, which is reserved for SML pipelines, while the "B" columns were used only for LLM-SML pipelines. The summary includes *macro* $F_1$ scores across the lowest, mean, and highest values and the *evaluation time*. The table also records the number of pipeline evaluations that ended up in timeouts and the ones exceeding either the RAM or the VRAM limits, alongside the total pipeline evaluations for each experiment.

The results indicate that SML pipelines achieved the highest *macro* $F1$ performance across all tasks during training. However, LLM-SML pipelines' lowest and mean *macro* $F1$ performance were consistently better than their counterparts. Regarding *evaluation Time*, the lowest, mean, and highest values of Setup A were slightly better than those of Setup B, except for the IMDB task, where we see an outlier. This is reflected in the number of pipelines discovered. For instance, in IMDB, we discovered more than twice as many SML pipelines as LLM-SML pipelines in the same time frame.

Table 7 compares the best-performing pipelines and the state-of-the-art solutions across the target tasks. The SOTA models outperformed the automatically discovered pipelines in all cases, with the most considerable difference observed in the Yelp Reviews Full task. The SML pipelines were consistently more environmentally friendly than their LLM-SML counterparts. For instance, IMDB_A_3 emitted almost ninety times less carbon dioxide than IMDB_B_1 while achieving higher performance.

Furthermore, Figure 2 illustrates the correlation between accuracy and carbon emissions for each task's pipelines in the Pareto fronts. We can observe a trend in which

| Measure | | IMDB (A) | IMDB (B) | AG (A) | AG (B) | YR (A) | YR (B) |
|---|---|---|---|---|---|---|---|
| *macro* $F_1$ | Lowest | 0.22 | **0.33** | 0.01 | **0.09** | 0.00 | **0.15** |
| | Mean | 0.54 | **0.61** | 0.37 | **0.67** | 0.19 | **0.31** |
| | Highest | **0.90** | 0.88 | **0.91** | 0.90 | **0.51** | 0.48 |
| *evaluation Time* | Lowest | **2.537** | 50.32 | **2.738** | 202.4 | **43.34** | 1348 |
| | Mean | **130.4** | 336.9 | **40.69** | 595.5 | **90.32** | 2002 |
| | Highest | 3537 | **975.9** | **146.9** | 1173 | **229.2** | 2931 |
| Timeouts Amount | | 24 | 25 | 104 | 32 | 39 | 38 |
| RAM exceedance Amount | | 59 | 6 | 128 | 1 | 73 | 1 |
| VRAM exceedance Amount | | 0 | 156 | 0 | 14 | 0 | 14 |
| Total Pipeline Evaluations | | 608 | 270 | 489 | 71 | 129 | 56 |

Table 6: Performance and computational efficiency metrics during pipeline discovery (Training). Total Pipeline Evaluations include successfully evaluated pipelines and pipelines that exceeded resource limits. Pipelines that failed to evaluate due to invalid parameter configuration or runtime errors were excluded.

| Dataset | Type | Solution Id | Acc. | Time (s) | Energy (kWh) | Emissions ($gCO_2$) |
|---|---|---|---|---|---|---|
| IMDB | SML | IMDB_A_1 | 0.887 | 1037s | 0.144 | 27.93 |
| | | IMDB_A_2 | 0.882 | 18s | 2.41e-3 | 0.4680 |
| | | IMDB_A_3 | 0.878 | **10s** | **1.31e-3** | **0.255** |
| | LLM-SML | IMDB_B_1 (Electra base) | 0.870 | 601s | 0.119 | 23.24 |
| | | IMDB_B_2 (DistilBERT) | 0.865 | 240s | 5.05e-2 | 9.795 |
| | LLM | DistilBERT* | 0.854 | - | - | - |
| | | DistilBERT** | 0.924 | - | - | - |
| | | DistilBERT† | - | 341s | 0.045 | 12 |
| | | BERT** | 0.936 | - | - | - |
| | | BERT† | - | 478s | 0.062 | 19 |
| | | BERT-ITPT-FiT (Sun et al., 2019) | 0.956 | - | - | - |
| | | XLNet (Yang et al., 2019) | **0.968** | - | - | - |
| AG | SML | AG_A_1 | 0.916 | **219s** | **2.95e-2** | **5.722** |
| | | AG_A_2 | 0.911 | 1106s | 0.153 | 29.80 |
| | LLM-SML | AG_B_1 (BERT base) | 0.901 | 1058s | 0.173 | 33.63 |
| | LLM | XLNet (Yang et al., 2019) | 0.955 | - | - | - |
| | | RoBERTa-GCN (Lin et al., 2021) | 0.956 | - | - | - |
| | | CARP (Sun et al., 2023) | **0.964** | - | - | - |
| YR-F | SML | YR_A_1 | 0.530 | **77s** | **9.17e-3** | **1.7783** |
| | LLM-SML | YR_B_1 (Electra small) | 0.496 | 2119s | 71.42 | 76.49 |
| | LLM | BigBird (Zaheer et al., 2020) | 0.721 | - | - | - |
| | | XLNet (Yang et al., 2019) | **0.729** | - | - | - |

Table 7: Comparison of Selected Pipelines with State-of-the-Art Solutions: Accuracy, Time, Energy, and Carbon Emissions. The table shows the performance of selected pipelines with an accuracy within 0.01 of the highest-performing pipelines across different datasets, along with the employed LLM for hybrid pipelines. The report includes information on accuracy, training and testing time, energy consumption, and carbon emissions. Data for entries marked with † was extracted from Wang et al. (2023) (Appendix A.3), obtained by fine-tuning these models in an A100-based machine (4xA100 GPUs + 32 Intel Xeon processors). Values for entries marked with ** were extracted from Pipalia, Bhadja, and Shukla (2020). Accuracy for entry marked with * was extracted from Ng et al. (2023), which only trained the classification head.

SML pipelines are more carbon-friendly while achieving comparable performance results. On average, SML solutions in the Pareto fronts were over five times more carbon-efficient than their counterparts.

## 5 Discussion

Regarding the AutoML search process, Figure 3 illustrates how the system's optimization strategy attempted to balance out *macro* $F_1$ and *evaluation time* across iterations in IMDB As shown in Table 6, SML pipelines showed higher top performance and less training time on average than their hybrid counterparts, which translated to better accuracy and carbon efficiency on the test sets (see Figure 2). This highlights the potential of AutoML and multi-objective optimization for automatically developing greener ML solutions, eventually leading to an environmentally safe democratization of Machine Learning.

As shown by Table 7, none of the pipelines we produced matched the performance of state-of-the-art (SOTA) solutions. However, regarding carbon emissions, our solutions are consistently and significantly more environmentally friendly than the SOTA models.

On the IMDB dataset, our DistilBERT pipeline resulted in 20% and 50% less carbon emissions than fine-tuned DistilBERT and BERT (Wang et al., 2023), respectively. Our greener solution (MR_A_3), in turn, is about 47 and 74 times more carbon-efficient than both fine-tuned models, respectively, while also outperforming our DistilBERT pipeline.

Being both BERT and DistilBERT quite popular, there is a great deal of available data regarding their performance. Pipalia, Bhadja, and Shukla (2020) reported 92.5% and 93.6% accuracy for finetuned DistilBERT and BERT, respectively, on the IMDB dataset. Although they do not report carbon emissions nor energy consumed by their training, we assume their actual emissions are similar or worse than the reported by Wang et al. (2023), which employed highly optimized equipment (Wang et al. (2023) also tested on lesser optimized hardware, rendering worse measurements). On the other hand, Ng et al. (2023) trained a single classification head connected to the BERT embeddings, obtaining an accuracy of 85.4%, worse than our models.

Although we do not have concrete data regarding the SOTA models' emissions, we estimate that the generated DistilBERT pipeline produced fewer emissions than BERT-ITPT-Fit would on the same energy infrastructure as with BERT. Although we cannot establish a scaling relationship between BERT and XL-Net, we assume the relationship applies due to their similar size. However, this only serves as an empirical estimate and not a definitive proof.

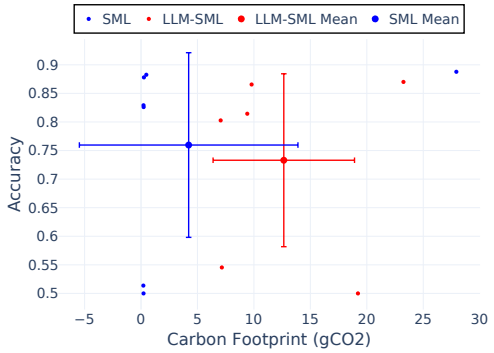
Regarding the visited pipelines, figure 4 shows all LLMs that participated in valid solutions generated by the optimization process. Almost every model from Table 2 was evaluated at least once in the three tasks. DistilBERT and Electra (Clark et al., 2020) variants were the most commonly used models in the experiments, while generative models such as GPT-2 (Radford et al., 2019) and T5 (Raffel et al., 2020) were the least used. However, BERT, Roberta, and DistilBERT had the highest mean *macro* $F_1$ scores. Concerning evaluation time, Distilbert, BERT, AlBERT (Lan et al., 2019), and Electra were the most efficient.

The results presented in Table 6 are further supported by Figure 5. The figure shows that SML pipelines are more efficient in terms of compute time. Conversely, LLM-SML pipelines have better average values with less variance, even though they do not have the best *macro* $F_1$ scores. This demonstrates the potential of SML models to fill the gaps in AI-based applications as lightweight models for specific purposes. By using AutoML to compensate for the lack of flexibility of SML models, it is possible to use these models as building blocks to create complex and general systems.

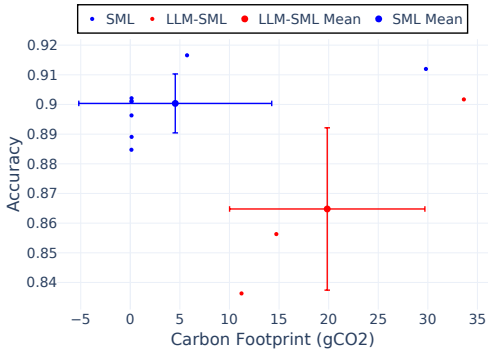
Our findings demonstrate that shallow machine-learning methods, paired with effective AutoML strategies, are competitive in specific NLP tasks. They also emphasize the importance of considering both performance and computational resource usage when designing solutions, especially in real-world applications where resources are limited. Given their higher resource consumption and larger carbon footprint, it is crucial not to overuse LLMs when more efficient and similarly performant alternatives exist.

## 6 Conclusions

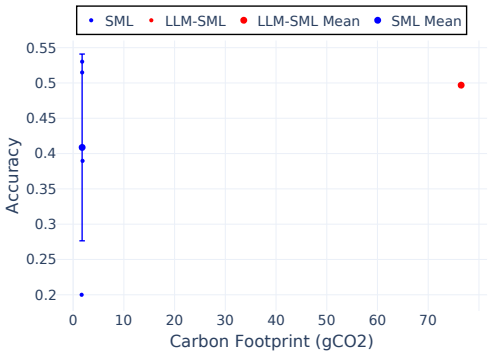
This paper explores the balance between accuracy and environmental sustainability in NLP, examining the resource efficiency of SML methods versus LLMs using AutoML.




(a) IMDb Movie Reviews

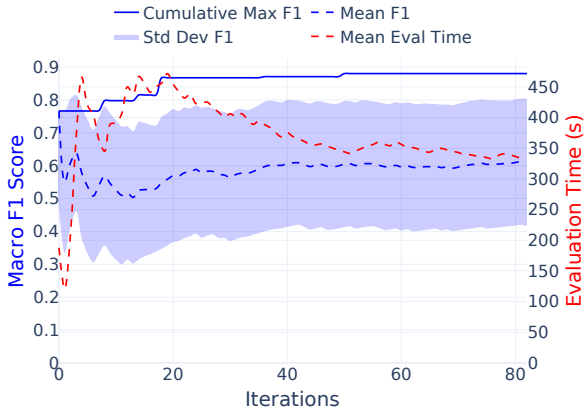



(b) AG News

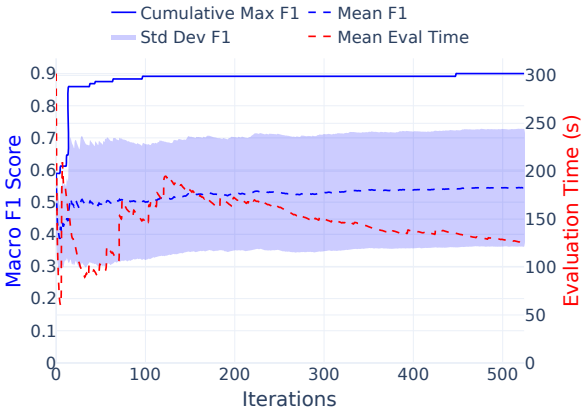



(c) Yelp Reviews Full

Figure 2: Comparative analysis of accuracy and carbon footprint of SML and LLM-SML pipelines Across Datasets. We report mean and standard deviation for both accuracy and carbon footprint.




(a) IMDB (LLM-SML)




(b) IMDB (SML)

Figure 3: Our proposal's optimization statistics regarding *macro* $F_1$ and *evaluation time* of the discovered pipelines in search. We report mean *macro* $F_1$, cumulative maximum *macro* $F_1$, standard deviation of *macro* $F_1$, and mean *evaluation time* over iterations (pipeline evaluations).

Our findings reveal that, although simpler pipelines don't necessarily surpass state-of-the-art models, they significantly mitigate environmental impact. The DistilBERT hybrid pipeline, for instance, reduces carbon emissions by 20% compared to fine-tuned DistilBERT models and by 50% relative to BERT models. We discovered SML optimal pipelines with competitive performance and up to 70 times less carbon emissions than hybrid or fully LLM pipelines, such as standard BERT and DistilBERT variants. This highlights a pivotal shift towards less reliance on resource-intensive LLMs for NLP tasks, with SML methods filling crucial performance gaps in a more eco-friendly manner. Our results underscore the untapped potential of SML techniques as sustainable alternatives to LLMs in

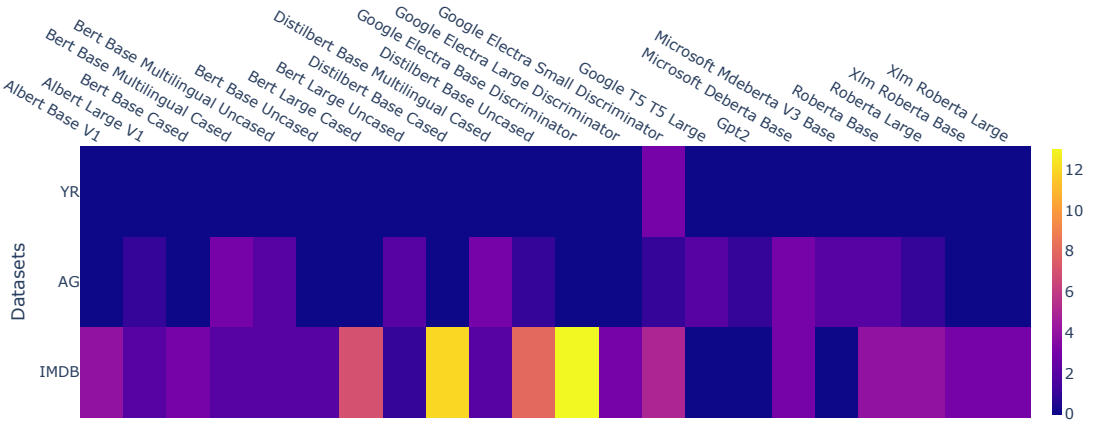



Figure 4: Occurrences of LLMs participating in solutions across tasks.

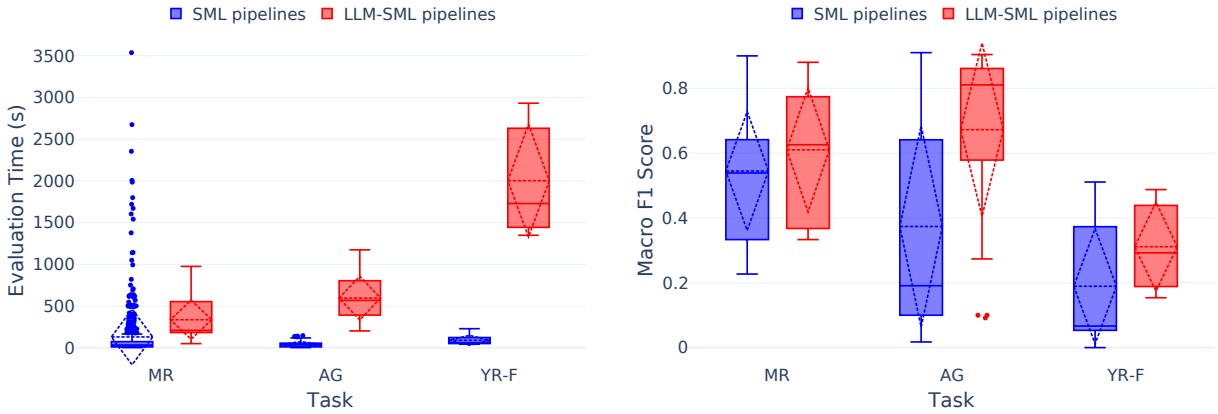



Figure 5: Comparison of SML and LLM-SML pipelines discovered. We report mean and standard deviation for *evaluation time* (a) and *macro* $F_1$ (a) on training.

specific NLP applications, further emphasizing AutoML's critical role in sculpting environmentally conscious AI models.

While our exploration yielded practical insights, it is crucial to contextualize these findings within the scope of the study's inherent limitations. Our reliance on several datasets introduces another limitation, potentially skewing our findings' applicability across the diverse landscape of NLP tasks. Including a broader array of datasets and a more diversified model selection, especially integrating fine-tuning capabilities into AutoML, beckons as the next frontier for research. Such expansions promise to unravel the nuanced dynamics between AI model performance, efficiency, and sustainability.

## *Acknowledgements*

This research has been partially funded by the University of Alicante and the University of Havana, the Spanish Ministry of Science and Innovation, the Generalitat Valenciana, and the European Regional Development Fund (ERDF) through the following funding: At the national level, the following projects were granted: COOLANG (PID2021-122263OB-C22); funded by MCIN/AEI/10.13039/501100011033 and, as appropriate, by "ERDF A way of making Europe", by the "European Union" or by the "European Union NextGenerationEU/PRTR". Also, the VIVES: "Pla de Tecnologies de la Llengua per al valencià" project (2022/TL22/00215334) from the Projecte Estratègic per a la Recuperació i Transformació Econòmica (PERTE). At regional level, the Generalitat Valenciana (Conselleria d'Educacio, Investigacio, Cultura i Esport), granted funding for NL4DISMIS (CIPROM/2021/21).

## *References*


Anthony, L. F. W., B. Kanding, and R. Selvan. 2020. Carbontracker: Tracking and predicting the carbon footprint of training deep learning models. *arXiv preprint arXiv:2007.03051*.

Bannour, N., S. Ghannay, A. Névéol, and A.-L. Ligozat. 2021. Evaluating the carbon footprint of nlp methods: a survey and analysis of existing tools. In *Proceedings of the second workshop on simple and efficient natural language processing*, pages 11–21.

Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. 2020. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Chollet, F. 2018. Keras: The python deep learning library.

Chung, H. W., L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. 2022. Scaling instruction-finetuned language models.

Clark, K., M.-T. Luong, Q. V. Le, and C. D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Conneau, A., K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dodge, J., T. Prewitt, R. Tachet des Combes, E. Odmark, R. Schwartz, E. Strubell, A. S. Luccioni, N. A. Smith, N. DeCario, and W. Buchanan. 2022. Measuring the carbon intensity of ai in cloud instances. In *Proceedings of the 2022 ACM conference on fairness, accountability, and transparency*, pages 1877–1894.

Estévez-Velarde, S., Y. Gutiérrez, Y. Almeida-Cruz, and A. Montoyo. 2021. General-purpose hierarchical optimisation of machine learning pipelines with grammatical evolution. *Information Sciences*, 543:58–71.

Estevez-Velarde, S., Y. Gutiérrez, A. Montoyo, and Y. Almeida-Cruz. 2019. Automl strategy based on grammatical evolution: A case study about knowledge discovery from text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4356–4365.

Estevez-Velarde, S., Y. Gutiérrez, A. Montoyo, and Y. A. Cruz. 2020. Automatic discovery of heterogeneous machine learning pipelines: An application to natural language processing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3558–3568.

Faiz, A., S. Kaneda, R. Wang, R. Osi, P. Sharma, F. Chen, and L. Jiang. 2023. Llmcarbon: Modeling the end-to-end carbon footprint of large language models. *arXiv preprint arXiv:2309.14393*.

Fedus, W., B. Zoph, and N. Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Feurer, M., K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter. 2020. Autosklearn 2.0: The next generation. *arXiv: Learning*.

Floridi, L. and M. Chiriatti. 2020. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694.

González-Carvajal, S. and E. C. Garrido-Merchán. 2020. Comparing bert against traditional machine learning text classification. *arXiv preprint arXiv:2005.13012*.

Gu, Y., L. Dong, F. Wei, and M. Huang. 2023. Minillm: Knowledge distillation of large

language models. In *The Twelfth International Conference on Learning Representations*.

He, P., J. Gao, and W. Chen. 2023. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

He, P., X. Liu, J. Gao, and W. Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Holmes, G., A. Donkin, and I. H. Witten. 1994. Weka: a machine learning workbench. pages 357–361. IEEE.

Honnibal, M., I. Montani, S. Van Landeghem, and A. Boyd. 2020. spacy: Industrial-strength natural language processing in python.

Hutter, F., L. Kotthoff, and J. Vanschoren. 2019. *Automated Machine Learning*. Springer.

Jin, H., Q. Song, and X. Hu. 2019. Auto-keras: An efficient neural architecture search system. pages 1946–1956. ACM.

Kaplan, J., S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Kotthoff, L., C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown. 2019. Auto-weka: Automatic model selection and hyperparameter optimization in weka. *Automated machine learning: methods, systems, challenges*, pages 81–95.

Kowsari, K., K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown. 2019. Text classification algorithms: A survey. *Information*, 10(4):150.

Lan, Z., M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

LeDell, E. and S. Poirier. 2020. H2o automl: Scalable automatic machine learning. In *Proceedings of the AutoML Workshop at ICML*, volume 2020.

Lepikhin, D., H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Lin, Y., Y. Meng, X. Sun, Q. Han, K. Kuang, J. Li, and F. Wu. 2021. Bertgcn: Transductive text classification by combining gcn and bert. *arXiv preprint arXiv:2105.05727*.

Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Loper, E. and S. Bird. 2002. Nltk: the natural language toolkit. *arXiv preprint cs/0205028*.

Maas, A. L., R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. 2011. Learning word vectors for sentiment analysis. In D. Lin, Y. Matsumoto, and R. Mihalcea, editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June. Association for Computational Linguistics.

Mohr, F., M. D. Wever, and E. Hüllermeier. 2018. Ml-plan: Automated machine learning via hierarchical planning. *Machine Learning*, 107(8):1495–1515.

Ng, S. Y., K. M. Lim, C. P. Lee, and J. Y. Lim. 2023. Sentiment analysis using distilbert. In *2023 IEEE 11th Conference on Systems, Process Control (ICSPC)*, pages 84–89.

Nori, H., N. King, S. M. McKinney, D. Carignan, and E. Horvitz. 2023. Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv:2303.13375*.

Olson, R. S. and J. H. Moore. 2019. Tpot: A tree-based pipeline optimization tool for automating machine learning. In *Automated Machine Learning*. Springer, pages 151–160.

OpenAI. 2023. Gpt-4 technical report. Technical report. arXiv:2303.08774.

Öztürk, E., F. Ferreira, H. Jomaa, L. Schmidt-Thieme, J. Grabocka, and F. Hutter. 2022. Zero-shot automl with pretrained models. In *International Conference on Machine Learning*, pages 17138–17155. PMLR.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Pipalia, K., R. Bhadja, and M. Shukla. 2020. Comparative analysis of different transformer based architectures used in sentiment analysis. In *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*, pages 411–415.

Qiu, X., T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. Language models are unsupervised multitask learners.

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Řehůřek, R. and P. Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA.

Rodola, G. 2020. Psutil documentation.

Romano, J. D., T. T. Le, W. Fu, and J. H. Moore. 2021. Tpot-nn: augmenting tree-based automated machine learning with neural network estimators. *Genetic Programming and Evolvable Machines*, 22:207–227.

Sanh, V., L. Debut, J. Chaumond, and T. Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Schmidt, V., K. Goyal, A. Joshi, B. Feld, L. Conell, N. Laskaris, D. Blank, J. Wilson, S. Friedler, and S. Luccioni. 2021. Codecarbon: estimate and track carbon emissions from machine learning computing. *Cited on*, 20.

Schwartz, R., J. Dodge, N. A. Smith, and O. Etzioni. 2019. Green ai.

Sun, C., X. Qiu, Y. Xu, and X. Huang. 2019. How to fine-tune bert for text classification? In *Chinese computational linguistics: 18th China national conference, CCL 2019, Kunming, China, October 18–20, 2019, proceedings 18*, pages 194–206. Springer.

Sun, X., X. Li, J. Li, F. Wu, S. Guo, T. Zhang, and G. Wang. 2023. Text classification via large language models. *arXiv preprint arXiv:2305.08377*.

Thompson, N. C., K. Greenewald, K. Lee, and G. F. Manso. 2021. Deep learning's diminishing returns: The cost of improvement is becoming unsustainable. *IEEE Spectrum*, 58(10):50–55.

Thornton, C., F. Hutter, H. H. Hoos, and K. Leyton-Brown. 2013. Auto-weka: combined selection and hyperparameter optimization of classification algorithms. pages 847–855. ACM.

Wang, X., C. Na, E. Strubell, S. Friedler, and S. Luccioni. 2023. Energy and carbon considerations of fine-tuning bert. *arXiv preprint arXiv:2311.10267*.

Yang, Z., Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Yun-tao, Z., G. Ling, and W. Yong-cheng. 2005. An improved tf-idf approach for text classification. *Journal of Zhejiang University-Science A*, 6(1):49–55.

Zaheer, M., G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

Zhang, X., J. Zhao, and Y. LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.