Session-Aware URL Access Control for Study Productivity

Jaehyeong Cho (2024148055)

1 Introduction

When I try to study seriously, it is very easy to open one "harmless" tab and then fall into a rabbit hole of YouTube, Steam, or random blogs. Traditional productivity tools usually rely on a large static blacklist of domains, but my actual tasks change from day to day. Sometimes YouTube or Wikipedia are exactly what I need; other times they are pure distraction. I wanted a small, controllable example of how an AI pipeline could help with this problem.

In this homework, I build a *session-aware URL classifier*. Given a short description of my current study session and the title of a candidate webpage, the system predicts whether it should **allow** opening the page or **block** it. I start from two naïve baselines: a strict domain-based blacklist and a trivial "always-block" strategy. Then I fine-tune a small pre-trained transformer model on a synthetic dataset that I constructed from my own planned tasks and curated URLs.

2 Task Definition

- **Task description:** Classify pairs of (Session Description, Webpage Title) into binary labels: {0: Block, 1: Allow}.
- **Motivation:** Static blacklists fail for two main reasons. First, they cannot handle "double-edged" domains (e.g., YouTube is useful for lectures but distracting for essays). Second, they are incomplete and cannot possibly cover every distraction on the internet. A context-aware system solves this by dynamically evaluating relevance.
- **Input / Output:**
  - *Input:* A text string combining the session goal and webpage title
  - *Output:* A binary label where 1 indicates the page is productive/allowed and 0 indicates it is distracting/blocked.
- **Success criteria:** A successful system must minimize False Positives (safely blocking distractions) for a reasonable cost of False Negatives (blocking useful sites). High Precision and reasonable F1-score imply such cases.

3 Methods

This section describes the baselines and the improved transformer-based pipeline.

3.1 Naïve Baseline

Method: Domain Blacklist

For the baseline, I maintain a curated set of "blacklisted" domains in a JSON file (e.g., `youtube.com`, `reddit.com`). The list blocks domains across all categories in the "Freedom" app except for "Search

Engines" and "News". For each URL, if its hostname appears in the blacklist, I predict **block** ($y = 0$); otherwise I predict **allow** ($y = 1$). This simulates a stricter version of many real browser extensions.

Why it is naïve

This method is **static** and **context-blind**. It assumes a domain is either "always good" or "always bad." It cannot comprehend that `youtube.com` might be necessary for one session but distracting for another. It effectively treats the internet as a fixed list of known safe/unsafe sites.

Likely failure modes

- **False Positives (Blocking useful content):** Useful resources hosted on entertainment domains are blocked. For example, a required lecture on YouTube for the "Deep Neural Networks" session is blocked because the domain `youtube.com` is blacklisted.
- **False Negatives (Allowing distractions):** Distracting sites that are not in the curated list slip through. For example, a random "Top 10 AI Tools" blog post might be allowed during a "Google Login implementation" session simply because the specific blog domain wasn't on the blacklist.

3.2 AI Pipeline

Models used

For the improved pipeline, I fine-tune a small pre-trained transformer, `distilbert-base-uncased`, using the Hugging Face `transformers` library. I attach a simple linear classification head on top of the [CLS]-like representation to predict the binary label.

Pipeline stages

The pipeline consists of the following stages:

1. **Preprocessing.** For each labeled example, I concatenate the session description and the page title into a single string:

   "Session: <session_text> [SEP] Page: <title>".

2. **Tokenization.** I tokenize the text with the DistilBERT tokenizer using padding and truncation to a max length of 128 tokens.
3. **Decision component.** The transformer's pooled output is passed through a linear layer to produce logits.
4. **Training.** I train for 6 epochs using the AdamW optimizer (LR 5×10^{-5}), employing early stopping based on validation loss to prevent overfitting on the small dataset.

Design choices and justification

I chose `distilbert-base-uncased` because it is lightweight enough to fine-tune on a small dataset but powerful enough to understand semantic relationships (e.g., relating "Deep Learning" to "Neural Network" titles). Encoding both texts in one sequence allows the model to learn the interaction between the session goal and the page content.

4 Experiments

4.1 Datasets

Source and Labeling

I built a synthetic dataset reflecting my actual study habits. I defined five session descriptions (e.g., "Deep Neural Network finals", "Cloud Computing") and curated 60 URLs ranging from global hubs (YouTube) to specific documentation (Terraform).

I created 300 examples by taking the Cartesian product of sessions and URLs. Labels were assigned based on relevance:

- **Allow (1):** If the URL is useful for the specific session (e.g., Terraform docs for Cloud) or is a useful global tool (e.g., Google).
- **Block (0):** If the URL is irrelevant or distracting for that session.

The dataset is imbalanced: 256 negative (block) and 44 positive (allow) examples.

Splits

I used stratified sampling to split the 300 examples with a similar "allow" rate:

- **Train:** 210 examples.
- **Validation:** 30 examples.
- **Test:** 60 examples (51 block, 9 allow).

4.2 Metrics

I evaluate using **Precision**, **Recall**, and **F1-score** for the positive (allow) class.

- **Precision** measures safety: of the pages allowed, how many were actually useful? (High precision means few distractions slipped through).
- **Recall** measures utility: of the useful pages, how many did we allow?
- **F1-score** balances the two.

Why not just Accuracy? Since the dataset is imbalanced (85% negative), a trivial strategy that blocks *every* page would already achieve 85% accuracy. However, such a model is useless.

4.3 Results

Quantitative Comparison

The table below compares the naive blacklist baseline and the AI pipeline on the test set (N=60).

Method	Accuracy	Precision	Recall	F1-Score
Baseline (Blacklist)	0.25	0.14	**0.78**	0.24
AI Pipeline (Ours)	**0.88**	**0.67**	0.44	**0.53**

Analysis

Comparison to Blacklist: The blacklist achieves high recall (0.78) but fails catastrophically on precision (0.14). It essentially allows almost everything, letting through 43 false positives (distractions). The AI pipeline improves precision by nearly 5×, demonstrating it is much effective at filtering distractions while still identifying useful content.

Qualitative examples

Context-Aware Success (Model > Blacklist). For **Session 2** ("Studying for Deep Neural Network finals"), the user visited **YouTube** (https://www.youtube.com/).

- **Baseline:** Blocked ($y = 0$) because `youtube.com` is on the static blacklist.
- **Model:** Correctly **Allowed** ($y = 1$). The model likely associated the session context ("studying", "finals") with the utility of video lectures, overcoming the domain bias.

Filtering Success (Model > Blacklist). For **Session 1** ("Implement Google Login"), the user visited a blog titled **"10 Best AI Tools for Students"**.

- **Baseline:** Allowed ($y = 1$) because the domain `aeccglobal.com` is not blacklisted.
- **Model:** Correctly **Blocked** ($y = 0$). The model recognized that general "AI tools" are irrelevant to the specific task of implementing "Google Login," effectively filtering out productive procrastination.

Failure Mode: False Negatives. For **Session 3** ("Cloud Computing finals... Terraform IaC"), the user visited the documentation page **"Infrastructure as Code | Terraform"**.

- **Baseline:** Allowed ($y = 1$).
- **Model:** Incorrectly **Blocked** ($y = 0$). Despite the clear keyword match ("Terraform"), the model was overly conservative. This confirms the low Recall score observed in the quantitative metrics—the model prefers to block when in doubt.

5 Reflection and Limitations

The AI pipeline's performance was surprisingly high given the small, synthetic nature of the dataset. Since I was the sole labeler, the data inherently reflects my personal bias on what constitutes a "distraction," yet the model generalized well enough to outperform the blacklist significantly.

Regarding metrics, prioritizing **Precision** and **F1-score** proved to be the correct choice. Since the system is designed to be a "focus gate," minimizing False Positives (allowing distractions) is critical. However, relying solely on these metrics obscures whether the model truly reasoned about session alignment or simply memorized keywords. We cannot confirm if the model understood the session goal or just flagged specific terms.

This lack of interpretability is a key weakness. A future improvement would be to integrate a **reasoning generator** (e.g., an LLM) to output *why* a page was allowed, enabling qualitative cross-evaluation. Additionally, the model's notable conservativeness could be mitigated via **Active Learning**: implementing a user feedback loop where "wrongly blocked" pages are flagged by the user and added to the training set, allowing the model to adapt its decision boundary over time.