近期文章分享

南方科技大学，计算机科学与工程系
计算语言学与意识科学实验室
Computational Linguistics and Consciousness Science Lab
2024年11月8日
徐炀

Papers

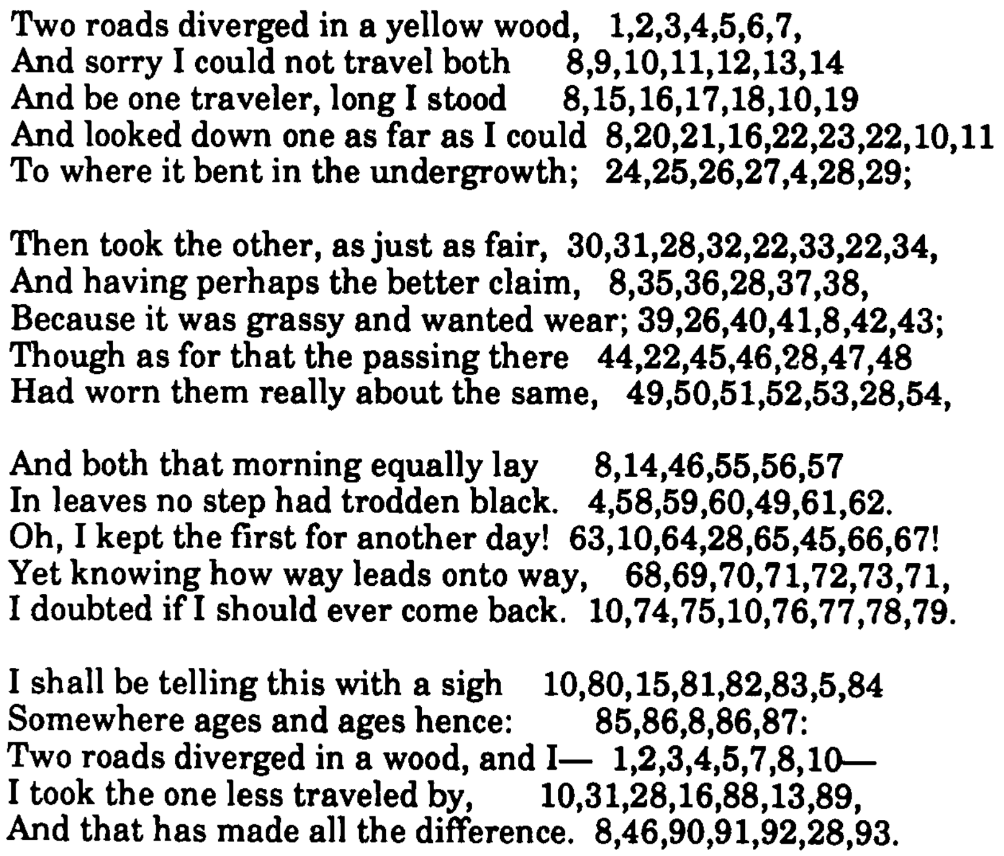
- A.F. Badalamenti et al.,
  - Lawful Systems Dynamics in How Poets Choose Their Words (1994)
  - Poisson Evolution in Word Selection (1994)
- Bao et al., (Author of Fast-DetectGPT)
  - White-Box Text Detectors using Proprietary LLMs: A Probability Distribution Estimation (PDE) Approach
- Nature watermark paper

A.F. Badalamenti et al.: Paper A +B

- Paper A: Lawful Systems Dynamics in How Poets Choose Their Words (1994) Behavioral Science

- The emergence of **new** words is a Poisson process
- Data: seven poets
- Two measures
  - ☐ waiting time between the invocation of new words
  - ☐ information entropy of word

Q1: Temporal property of new words?



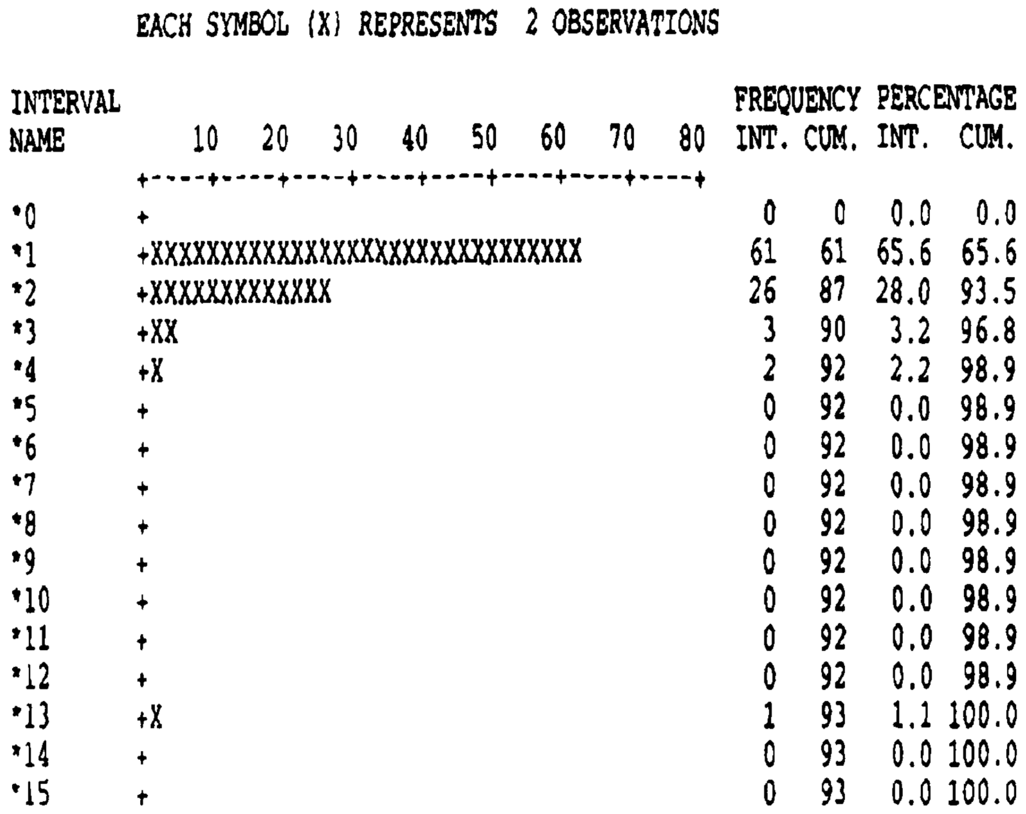
Two roads diverged in a yellow wood, 1,2,3,4,5,6,7,
And sorry I could not travel both 8,9,10,11,12,13,14
And be one traveler, long I stood 8,15,16,17,18,10,19
And looked down one as far as I could 8,20,21,16,22,23,22,10,11
To where it bent in the undergrowth; 24,25,26,27,4,28,29;

Then took the other, as just as fair, 30,31,28,32,22,33,22,34,
And having perhaps the better claim, 8,35,36,28,37,38,
Because it was grassy and wanted wear; 39,26,40,41,8,42,43;
Though as for that the passing there 44,22,45,46,28,47,48
Had worn them really about the same, 49,50,51,52,53,28,54,

And both that morning equally lay 8,14,46,55,56,57
In leaves no step had trodden black. 4,58,59,60,49,61,62.
Oh, I kept the first for another day! 63,10,64,28,65,45,66,67!
Yet knowing how way leads onto way, 68,69,70,71,72,73,71,
I doubted if I should ever come back. 10,74,75,10,76,77,78,79.

I shall be telling this with a sigh 10,80,15,81,82,83,5,84
Somewhere ages and ages hence: 85,86,8,86,87:
Two roads diverged in a wood, and I— 1,2,3,4,5,7,8,10—
I took the one less traveled by, 10,31,28,16,88,13,89,
And that has made all the difference. 8,46,90,91,92,28,93.

FIGURE 1
The Text of Frost's Poem, "The Road Not Taken" and Its Conversion into Numerical Data



```
                 EACH SYMBOL (X) REPRESENTS  2 OBSERVATIONS

INTERVAL                                                          FREQUENCY PERCENTAGE
NAME          10   20   30   40   50   60   70   80             INT. CUM. INT.  CUM.
         +----+----+----+----+----+----+----+----+
*0       +                                                          0    0  0.0   0.0
*1       +XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX                           61   61 65.6  65.6
*2       +XXXXXXXXXXXXX                                            26   87 28.0  93.5
*3       +XX                                                        3   90  3.2  96.8
*4       +X                                                         2   92  2.2  98.9
*5       +                                                          0   92  0.0  98.9
*6       +                                                          0   92  0.0  98.9
*7       +                                                          0   92  0.0  98.9
*8       +                                                          0   92  0.0  98.9
*9       +                                                          0   92  0.0  98.9
*10      +                                                          0   92  0.0  98.9
*11      +                                                          0   92  0.0  98.9
*12      +                                                          0   92  0.0  98.9
*13      +X                                                         1   93  1.1 100.0
*14      +                                                          0   93  0.0 100.0
*15      +                                                          0   93  0.0 100.0
```

FIGURE 2
Histogram of Waiting Times Between New Words in Frost's Poem, "The Road Not Taken".

Note: the waiting time is the number of words between a new word and the next occurrence of a new word.

Poisson = Negative Exponential

- Page 67:
- “A Poisson model characterizes the instantaneous probability that an event, here a new word, will occur.”
- The chief signature of a Poisson process is that the distribution of waiting times between successive events is **negative exponential**.

That is, if $f(t)$ is the density for the amount of time (words)

between successive events, then $f(t) = \lambda e^{-\lambda t}$, for some $\lambda > 0$.

(between a new word and the next new word)

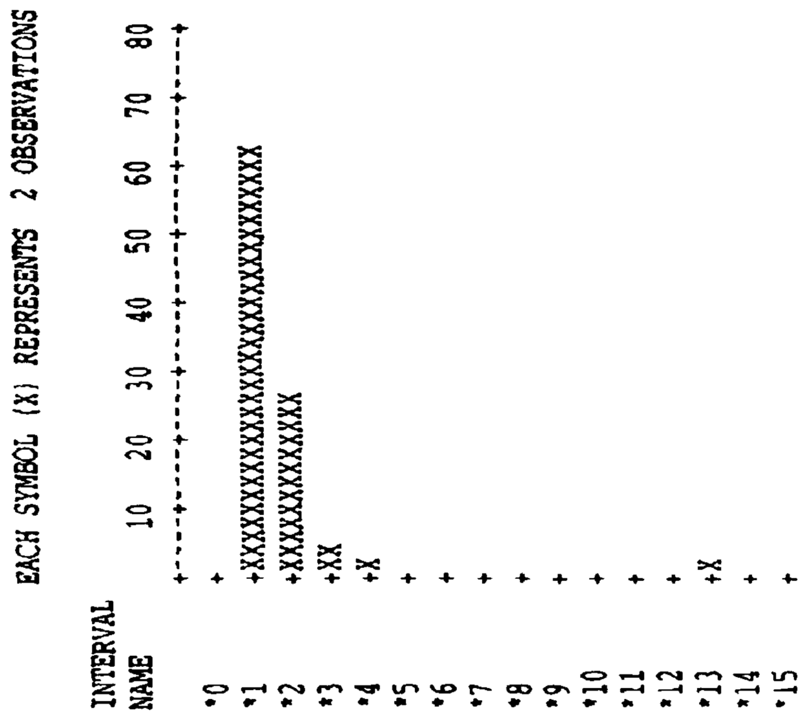


Figure 2
Histogram of Waiting Times Between New Words in Frost’s Poem, “The Road Not Taken”.

Note: the waiting time is the number of words between a new word and the next occurrence of a new word.

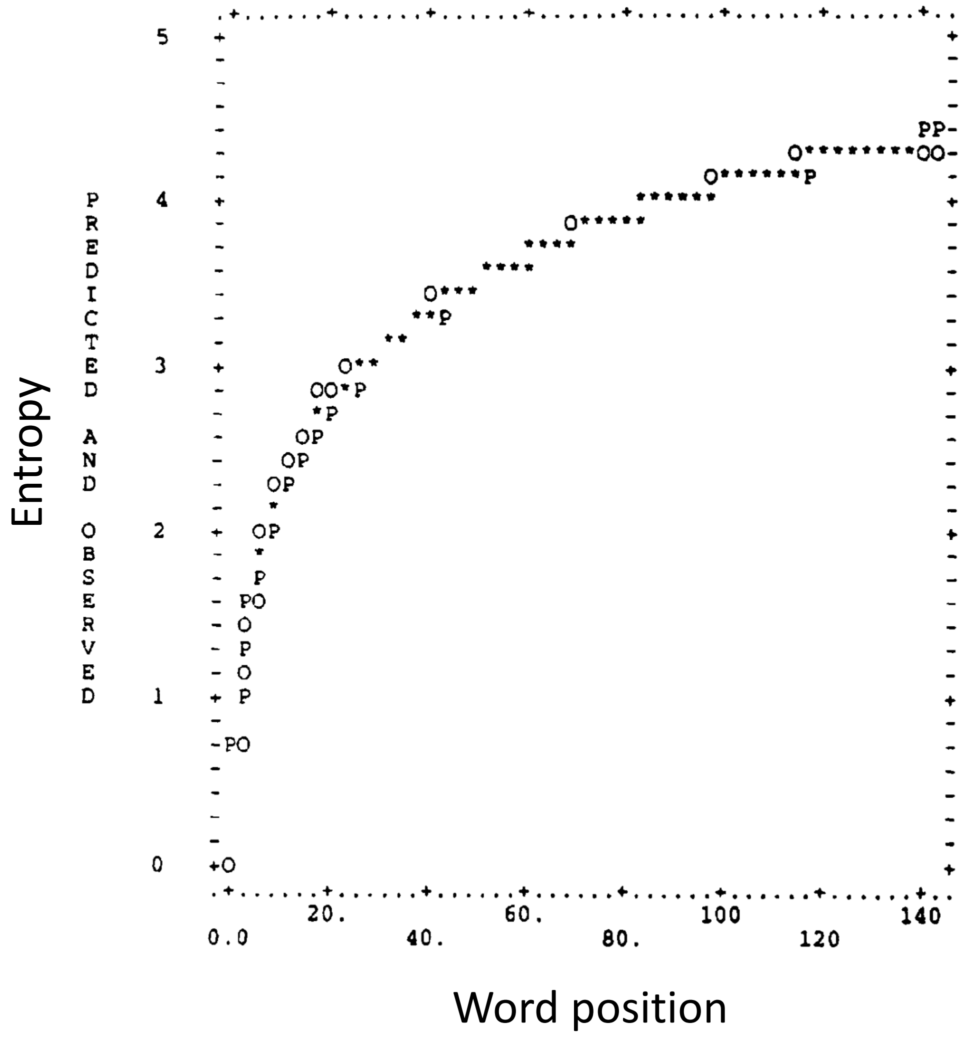
Q2: How does the variety of words grow?

- Define entropy using unigram probabilities

$$E(n) = -\sum_{i=1}^{i=2400} p_i \ln(p_i).$$

FIGURE 3
Fitted and Observed Entropy for New Words in Frost's Poem, "The Road Not Taken".

Note: O and P denote observed and fitted (Predicted) data, respectively; * denotes agreement of O and P to within the paper's resolution.



Paper B: Poisson Evolution in Word Selection

- The invocation of new words in human language samples is governed by a slowly changing Poisson process.
- The Poisson constant is in the form:

$$\lambda(t) = \lambda_1 (1 - \lambda_2 t) e^{-\lambda_2 t} + \lambda_3 (1 - \lambda_4 t) e^{-\lambda_4 t} + \lambda_5, \qquad \text{where } \lambda_i > 0, \quad i = 1, \ldots, 5.$$

- This form implies that there are **opening**, **middle** and **final** phases to the introduction of new words
- λ quantifies how the penchant (偏好) of humans to introduce new words declines with the progression of their narratives, written or spoken.

Poisson Evolution: $\lambda(t)$ evolves over time

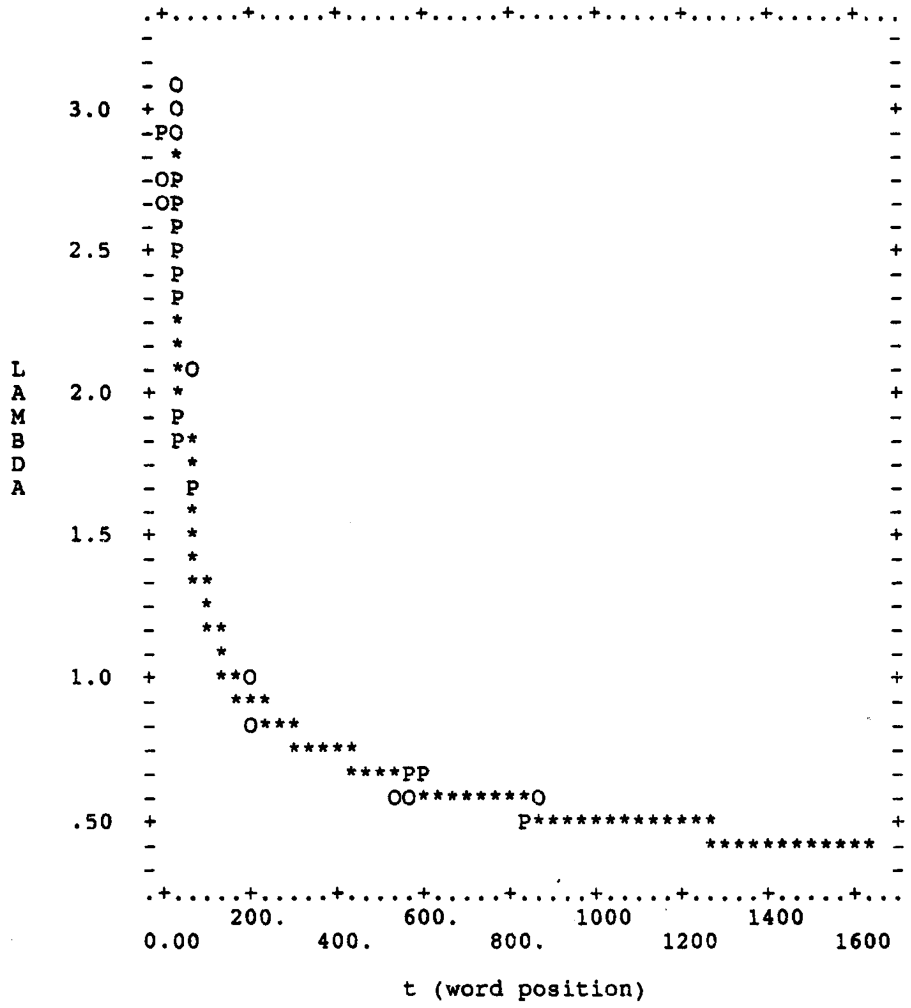


Figure 2. $\lambda_C(t)$ for Martin Luther King's speech. P denotes predicted by the regression. O denotes observed. $*$ denotes agreement of P and O to within plot resolution.

Fit Poisson constants

- Fit a cumulative rate constant $\lambda_C(t)$ from the data of previous graph:

$$\lambda_C(t) = \lambda_1 e^{-\lambda_2 t} + \lambda_3 e^{-\lambda_4 t} + \lambda_5, \qquad \text{where } \lambda i > 0, \quad i = 1, \ldots, 5.$$

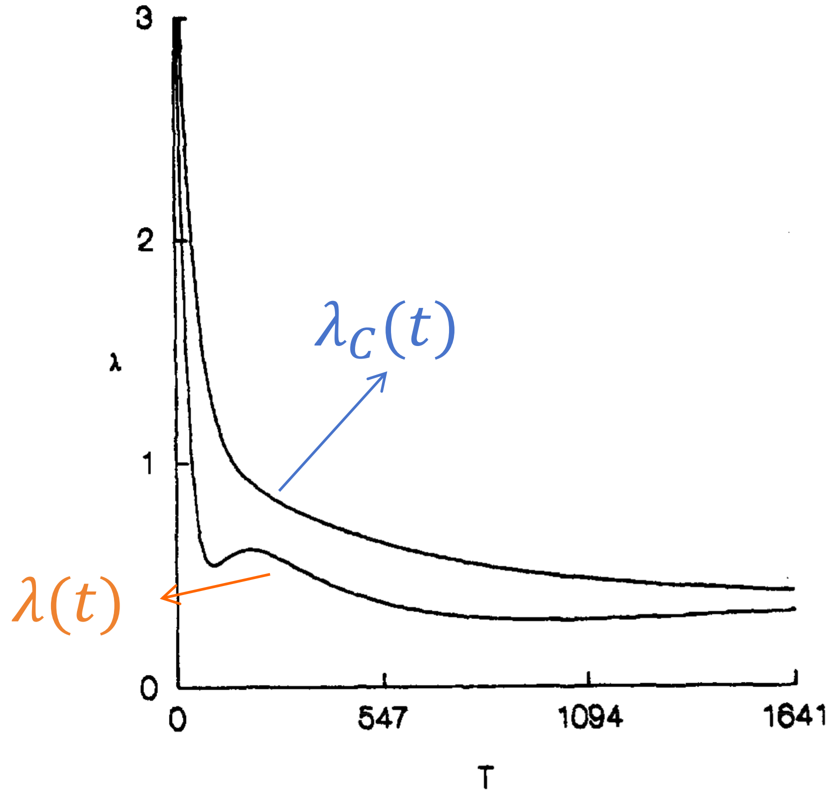
- Derive the true rate function $\lambda(t)$, using the differential equation (?) of above:

$$\lambda(t) = \lambda_1(1 - \lambda_2 t)e^{-\lambda_2 t} + \lambda_3(1 - \lambda_4 t)e^{-\lambda_4 t} + \lambda_5.$$

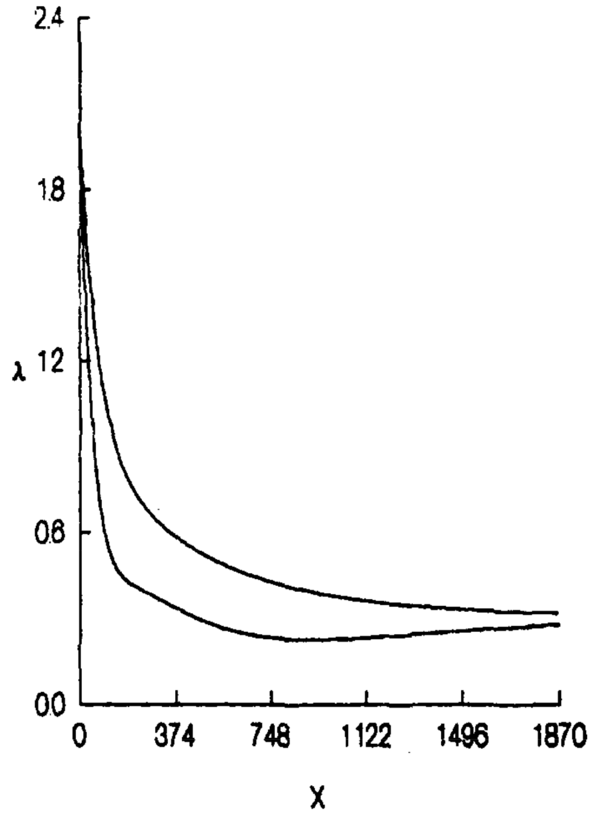
- Then the derivative of $\lambda(t)$ is:

$$\lambda'(t) = \lambda_1\lambda_2(-2 + \lambda_2 t)e^{-\lambda_2 t} + \lambda_3\lambda_4(-2 + \lambda_4 t)e^{-\lambda_4 t}.$$

Phases of new word generation



(a). The cumulative rate, $\lambda_C(t)$ and the true rate, $\lambda(t)$ for Martin Luther King's speech. $\lambda_C(t)$ is the upper plot.



(b). The cumulative rate, $\lambda_C(t)$ and the true rate, $\lambda(t)$ for Monologue 3. $\lambda_C(t)$ is the upper plot.

There is, thus, something of an overall U-shape to $\lambda(t)$. This is a familiar phenomenon in human task activity and is usually interpreted to signify entry into an activity with gusto, followed by fatigue, and concluding with fresh energy in anticipation of quitting the task.

Bao et al.: PDF Approach for Text Detection

- Propose Probability Distribution Estimation (PDE) to enable existing white-box methods to proprietary models

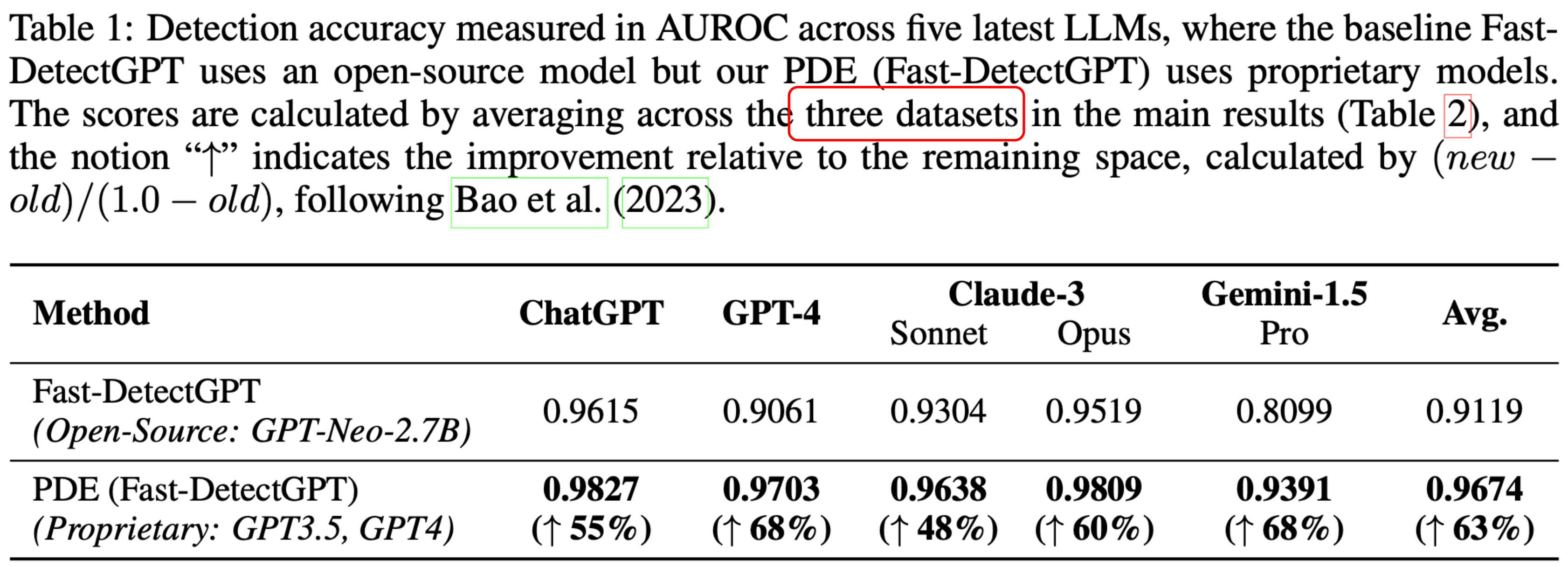
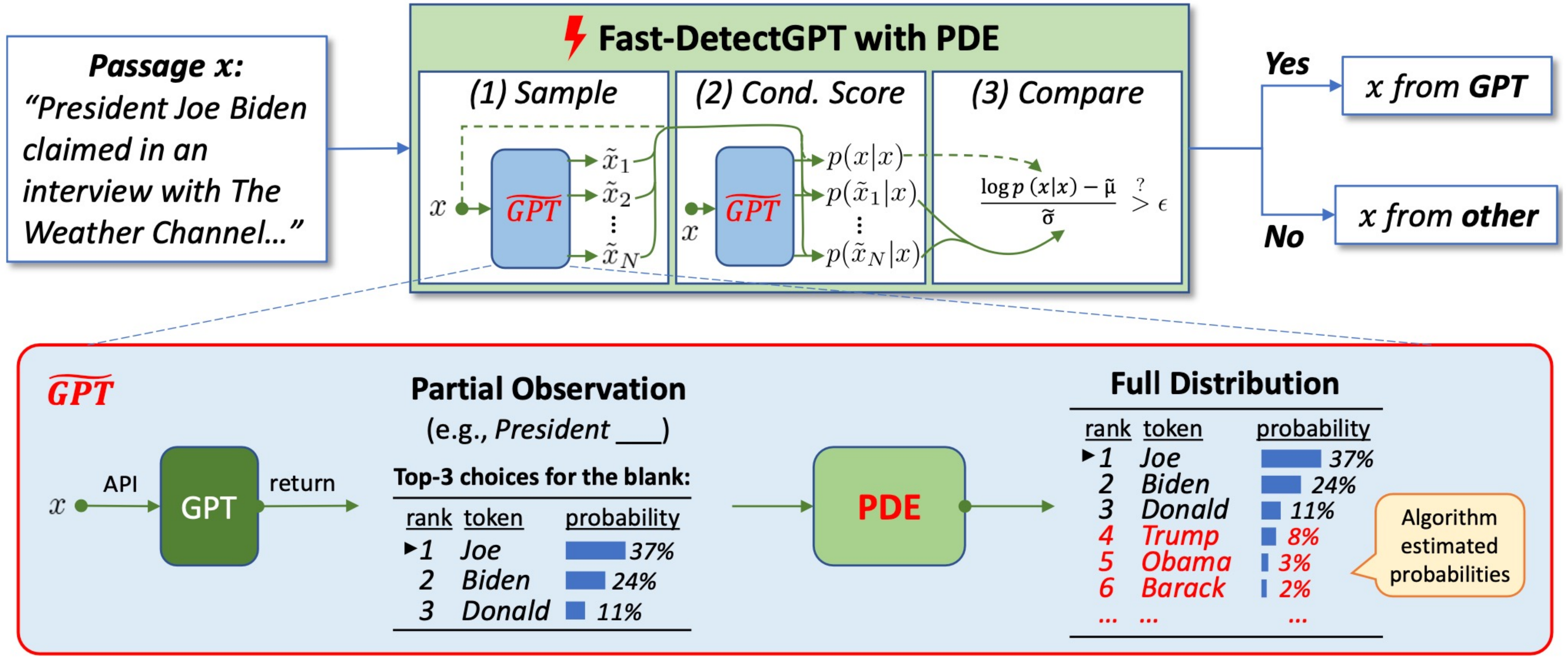


Table 1: Detection accuracy measured in AUROC across five latest LLMs, where the baseline Fast-DetectGPT uses an open-source model but our PDE (Fast-DetectGPT) uses proprietary models. The scores are calculated by averaging across the three datasets in the main results (Table 2), and the notion "↑" indicates the improvement relative to the remaining space, calculated by $(new - old)/(1.0 - old)$, following Bao et al. (2023).

Method	ChatGPT	GPT-4	Claude-3		Gemini-1.5	Avg.
			Sonnet	Opus	Pro	
Fast-DetectGPT *(Open-Source: GPT-Neo-2.7B)*	0.9615	0.9061	0.9304	0.9519	0.8099	0.9119
PDE (Fast-DetectGPT) *(Proprietary: GPT3.5, GPT4)*	**0.9827** (↑ **55%**)	**0.9703** (↑ **68%**)	**0.9638** (↑ **48%**)	**0.9809** (↑ **60%**)	**0.9391** (↑ **68%**)	**0.9674** (↑ **63%**)

Xsum, Writing, PubMed

Overview



Partial Observation (Top-3 choices for the blank):

rank	token	probability
▸1	Joe	37%
2	Biden	24%
3	Donald	11%

Full Distribution:

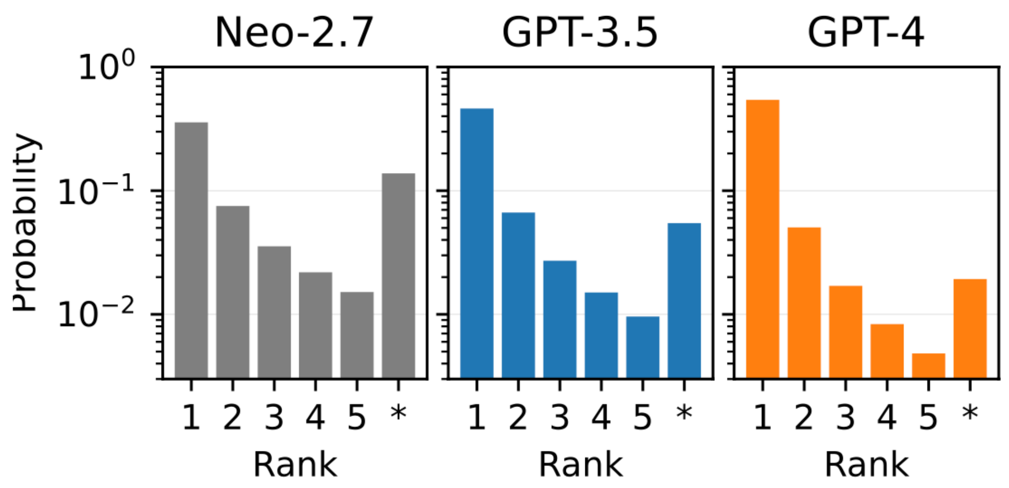
rank	token	probability
▸1	Joe	37%
2	Biden	24%
3	Donald	11%
4	Trump	8%
5	Obama	3%
6	Barack	2%
...	...	...

Figure 1: Fast-DetectGPT with *Probability Distribution Estimation (PDE)*. We take Fast-DetectGPT as an example to demonstrate how to apply white-box methods to proprietary models using PDE. $\widetilde{\text{GPT}}$ denotes the model with estimated distribution, where the partial observation (top-K probabilities) returned by the model API is completed into a full distribution. The '*token*' column is just for reference, which is not necessary for calculating the metric (conditional probability curvature).

Probability Distribution Estimation (PDE)

Formally, given a text sequence x, the proprietary model $p_\theta(\tilde{x}|x)$ provides us the likelihood $p_\theta(x_j|x_{<j})$ and the top-K token probabilities $p_\theta(\tilde{x}_j^k|x_{<j})|_{k=1}^{K}$ on each token position j, where k denotes the rank. The problem is then formulated as estimating $p_\theta(\tilde{x}_j|x_{<j})$ over the whole vocabulary according to the given information.

- The probability distribution follows a **decaying pattern**, where the larger models have a higher top-1 probability and a bigger decay factor



Probability Distribution Estimation (PDE)

- Three specific estimation algorithms
- 1. Using Geometric distribution: exponential decay with a fixed decay factor

$$p(k) = p_1 * (1 - p_1)^{k-1}, \text{ for } k \in [1..\infty],$$

- standard Geo. distr., where p_1 is the top-1 probability, with decay factor $\lambda = 1 - p_1$
- Consider near-Geo. distr.:

$$\begin{cases} p(k) = p_k, & \text{for } k \in [1..K] \\ p(k) = p_K * \lambda^{k-K}, & \text{for } k \in [K+1..M] \\ \sum_{k=1}^{M} p(k) = 1, & \end{cases}$$

decay factor λ is learned iteratively

Probability Distribution Estimation (PDE)

- 2. Using Zipfian Distribution
- Frequencies of words in natural languages usually adhere to Zipf’s law, where the word frequency –∝ word rank.（反比）

$$\begin{cases} p(k) = p_k, & \text{for } k \in [1..K] \\ p(k) = p_K * (\frac{\beta}{k-K+\beta})^{\alpha}, & \text{for } k \in [K+1..M] \\ \sum_{k=1}^{M} p(k) = 1, & \end{cases}$$

- α and β are positive params learned from a loss that minimizes their deviations from typical values.

Probability Distribution Estimation (PDE)

- 3. Using a Multi-Layer Perceptron (MLP) model, which accepts the top-K probabilities and predicts the probabilities for the rest of the ranks.

$$\begin{cases} p(k) = p_k, & \text{for } k \in [1..K] \\ p(k) = p_{\text{rest}} * p_{\text{MLP}_\theta}(k - K), & \text{for } k \in [K+1..M] \\ \sum_{k=1}^{M} p(k) = 1, & \end{cases}$$

where $p_{rest} = 1 - \sum_{k=1}^{K} p_k$ and p_{MLP_θ} represents the MLP predictive distribution.

PDE Outcomes

- MLP is closest to real distr.

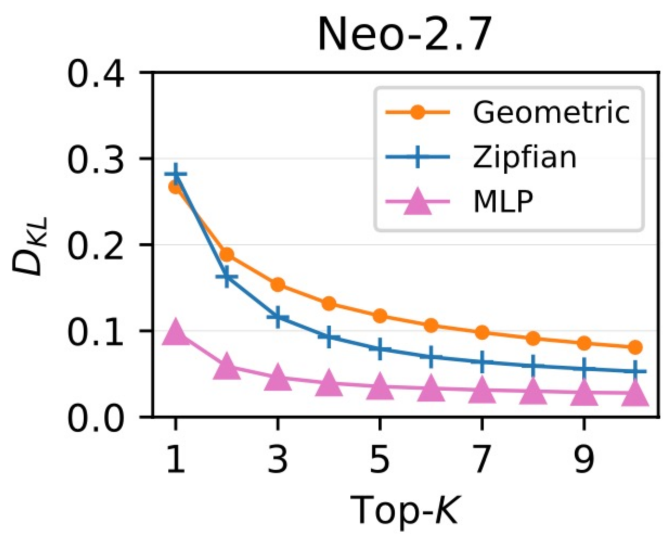
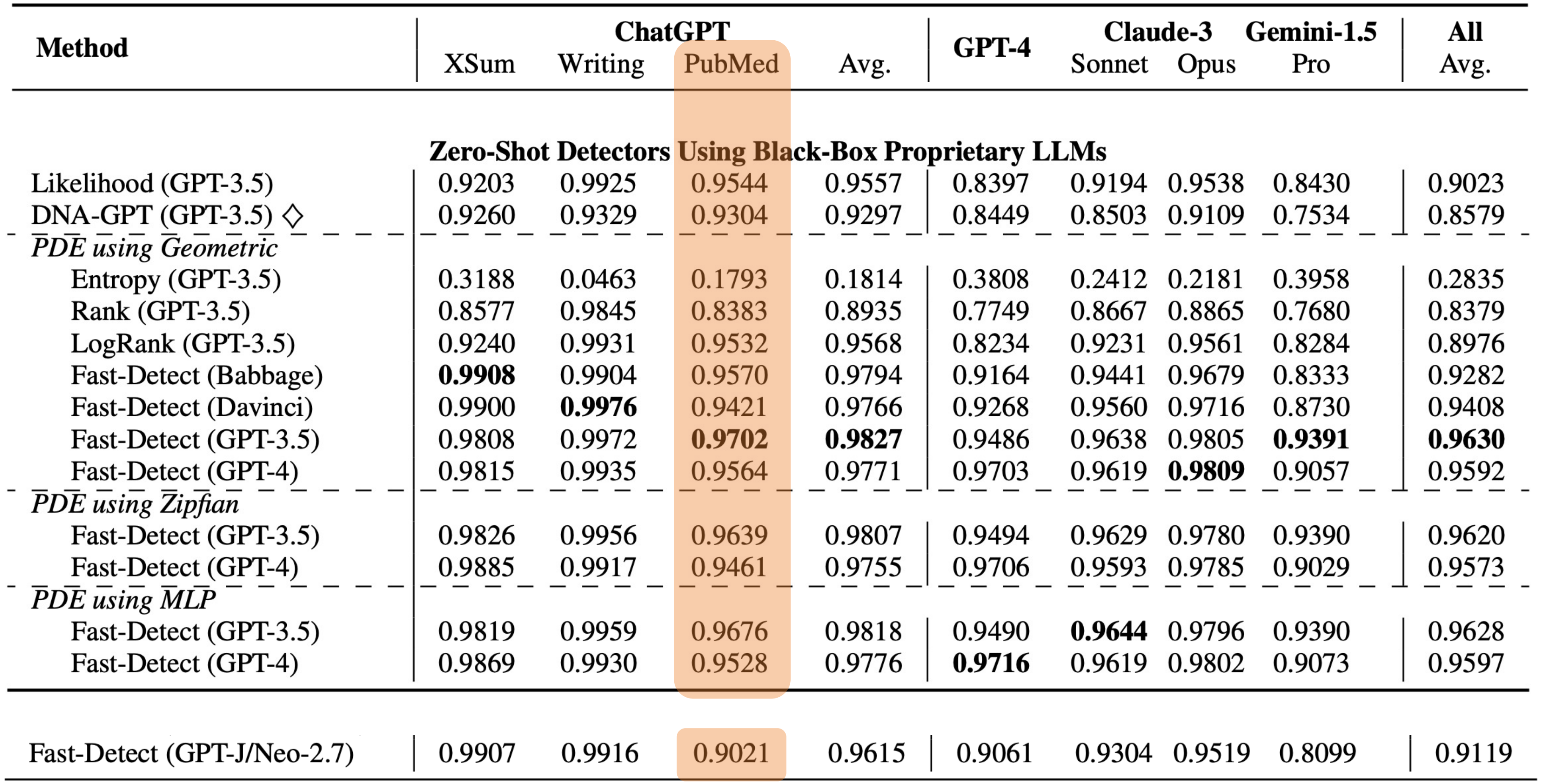


Figure 2: *KL divergence* against real distributions from Neo-2.7B.

Fast-Detect + PDE

PubMed: 短文本问答



Method	ChatGPT				GPT-4	Claude-3		Gemini-1.5	All
	XSum	Writing	PubMed	Avg.		Sonnet	Opus	Pro	Avg.
Zero-Shot Detectors Using Black-Box Proprietary LLMs									
Likelihood (GPT-3.5)	0.9203	0.9925	0.9544	0.9557	0.8397	0.9194	0.9538	0.8430	0.9023
DNA-GPT (GPT-3.5) ♢	0.9260	0.9329	0.9304	0.9297	0.8449	0.8503	0.9109	0.7534	0.8579
PDE using Geometric									
Entropy (GPT-3.5)	0.3188	0.0463	0.1793	0.1814	0.3808	0.2412	0.2181	0.3958	0.2835
Rank (GPT-3.5)	0.8577	0.9845	0.8383	0.8935	0.7749	0.8667	0.8865	0.7680	0.8379
LogRank (GPT-3.5)	0.9240	0.9931	0.9532	0.9568	0.8234	0.9231	0.9561	0.8284	0.8976
Fast-Detect (Babbage)	**0.9908**	0.9904	0.9570	0.9794	0.9164	0.9441	0.9679	0.8333	0.9282
Fast-Detect (Davinci)	0.9900	**0.9976**	0.9421	0.9766	0.9268	0.9560	0.9716	0.8730	0.9408
Fast-Detect (GPT-3.5)	0.9808	0.9972	**0.9702**	**0.9827**	0.9486	0.9638	0.9805	**0.9391**	**0.9630**
Fast-Detect (GPT-4)	0.9815	0.9935	0.9564	0.9771	0.9703	0.9619	**0.9809**	0.9057	0.9592
PDE using Zipfian									
Fast-Detect (GPT-3.5)	0.9826	0.9956	0.9639	0.9807	0.9494	0.9629	0.9780	0.9390	0.9620
Fast-Detect (GPT-4)	0.9885	0.9917	0.9461	0.9755	0.9706	0.9593	0.9785	0.9029	0.9573
PDE using MLP									
Fast-Detect (GPT-3.5)	0.9819	0.9959	0.9676	0.9818	0.9490	**0.9644**	0.9796	0.9390	0.9628
Fast-Detect (GPT-4)	0.9869	0.9930	0.9528	0.9776	**0.9716**	0.9619	0.9802	0.9073	0.9597
Fast-Detect (GPT-J/Neo-2.7)	0.9907	0.9916	0.9021	0.9615	0.9061	0.9304	0.9519	0.8099	0.9119

Previous SOTA

Advantages of PDE

- Universal 通用性

  PDE can also be used by other zero-shot detection methods like Entropy, Rank, and LogRank.

- Accurate and fast

Questions and Issues

- Best way to share slides? Ideas?