Name Generation with Autoregressive Character-level Language Modeling

Eljan Mahammadli

Prof. Steve Kaisler, Prof. Jamaladdin Hasanov

6917

03/08/2023

Project Objective

- Character-based language model for word generation.
- Comprehensive Study
- Multiple models: Bigram, MLP, Wavenet, RNN, GRU, and Transformer (GPT-2).
- Encapsulation with a command-line interface.
- Experimenting models on a dataset of company names.
- Versatile tool

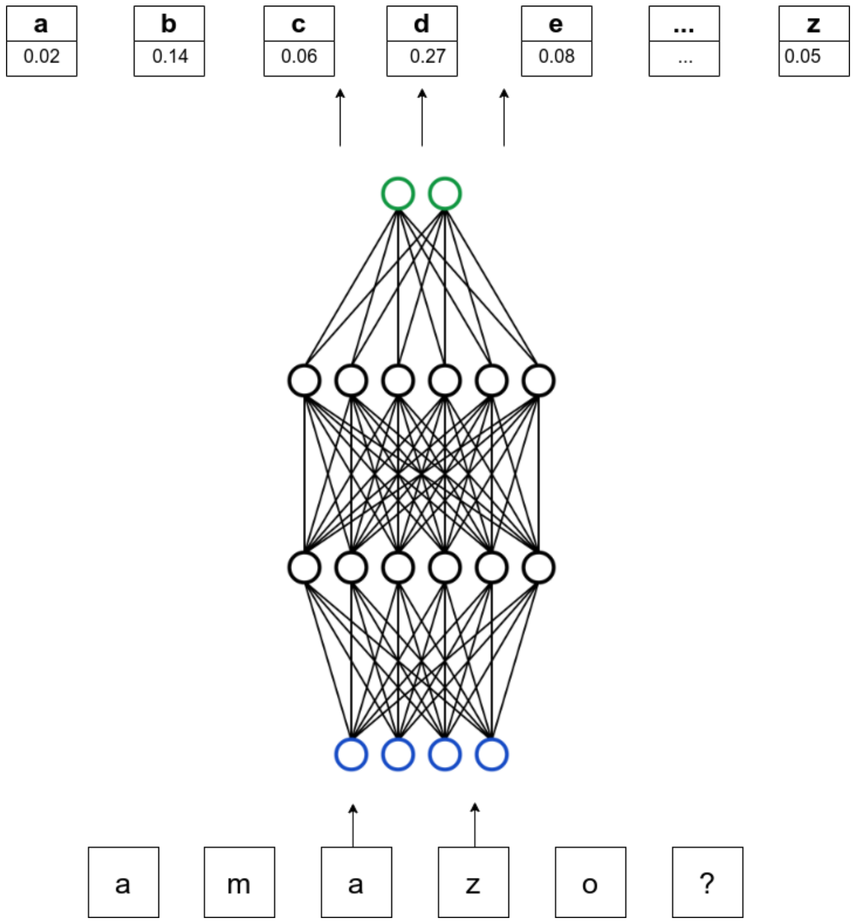
Heilmeier Questions

- What are you trying to do?
- How is it done today, and what are the limits of current practice?
- What is new in your approach and why do you think it is successful?
- Who cares?
- What are the risks and payoffs?
- How much did it cost?
- How long did it take?
- What are the midterm and final exams to check for success?

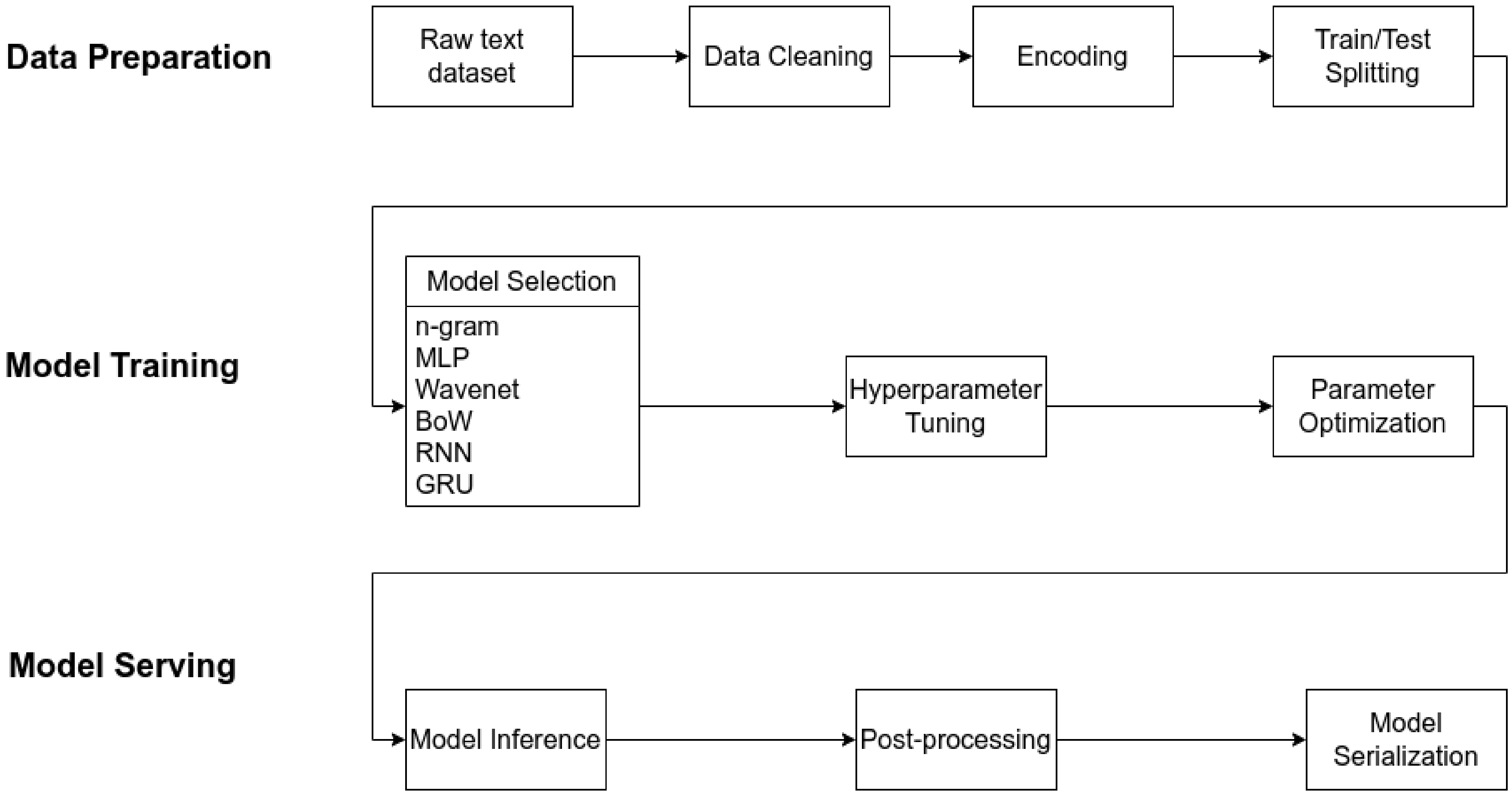
What is Character-level Language Model?

- Language models can predict the next token in a sequence.
- Probability is typically dependent on the preceding n tokens.
- Generate novel and valid names.
- Words often don't follow grammatical rules, so these models a flexible choice.

$P(w_1, w_2, \ldots, w_T) = \prod P(w_t \mid w_1, \ldots, w_{t-1})$ for t=1 to T



System Architecture



Key steps

Explored, developed, and conducted analyses of pivotal models and architectures that have significantly influenced the evolution of language modelling.

- N-gram, 1970s for LM
- MLP, Bengio et al. 2003
- CNN, DeepMind WaveNet 2016
- RNN, Mikolov et al. 2010
- GRU, Kyunghyun Cho et al. 2014
- Transformer, Vaswani et al. 2017

Model Specs

- **Bigram** - probability distribution of pairs of consecutive characters
- **MLP** - 10-d feature vector, 200 hidden neurons, 3 block size, 11 897 total parameters.
- **Wavenet** - 24-d character embedding space, 128 neurons in each hidden layer, 76 579 total parameters
- **RNN** - 4 layers, and 64 nodes per layer and hidden neurons, 11 803 total params
- **GRU** - same as RNN, 28 315 total parameters
- **Transformer** - 200K total parameters

What is innovative about the research?

- Comparison Across Architectures
- Model Adaptation
- Versatile Tool
- User-friendly Interface
- Flexible and Scalable Design

Command-line Interface

Users can easily input a name list and customize model behavior through optional arguments.

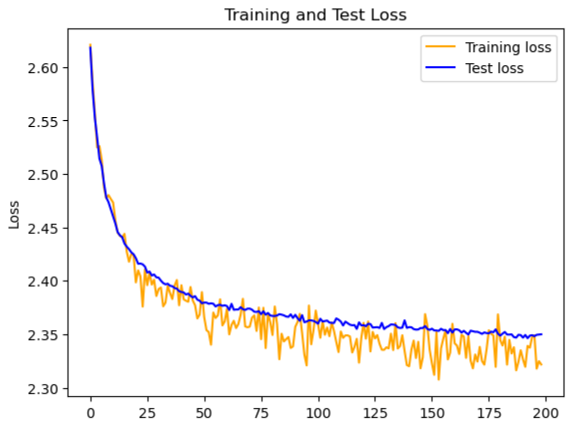
- **Input/output:** --input-dir, --output-dir, --resume, --inference, etc.
- **Model Configurations:** --model, --n-layer, --n-embd, …
- **Optimizations:** --learning-rate, --batch-size, --weight-decay
- etc

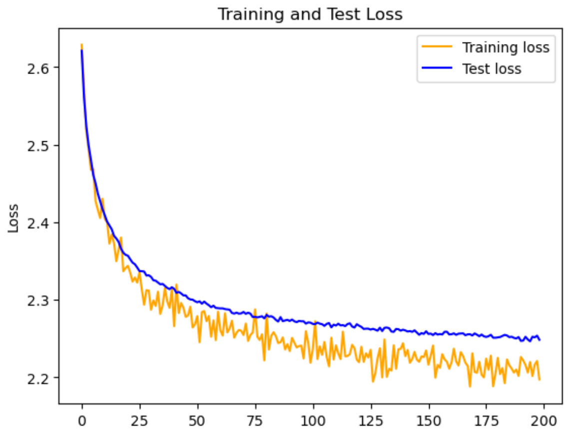
Example commands:

- $ python3 main.py -i dinosaurs.csv -o output -model transformer -n-head 4
- $ python3 main.py -i names.txt -o output --inference

Results

	Bigram	Trigram	MLP	WaveNet	RNN	GRU	Transformer
Loss	2.725	2.496	2.363	2.213	2.1814	2.1372	2.0695
Inference	paruis, joa, ftrtx, ts, halloum	tics, nutelamic, prel, tovil, reelesto	rid, forcend, welluma, cloudson, rantown	socience, homeline, keyibas, intellavids, alphars	bantist, talense, cooco, webtue, revicore	saitway, wineta, legomain, techips, creetap	techboundry, playmax, fisions, lightsoft, spreetware





PLACEHOLDER FOR TRANSFORMER LOSS FUNCTION

Loss Functions on train/test sets for RNN, GRU and Transformer respectively.

Training on small corpus of Azerbaijani names (for fun)

- **Loss:**
  - train: 1.5607 test: 2.0848
- Cangül
- Elmizə
- Sərban
- Rəyalə
- Gəlincam
- Timayət
- Dəryac
- Çeşmibəyid
- Xudalba
- Mudafər
- Abadətdin
- Nakizə
- Sərzad
- Qatibə
- Rafimə
- Gövdül
- Salibə

Conclusion

- Bottlenecks of each architecture.
- Wide choice of models.
- The system can be used to train on various domains/languages.
- Allows non-experts to leverage the power of language modelling for creative tasks.

Future Work

More recent or complex language models

Fine-tune existing models

Expand Dataset Diversity

Interface Improvement

Deployment and Scalability

Thanks for your attention!

Any questions?