cortical.io

# HIGH EFFICIENCY AI

4300x Increase in Energy Efficiency for AI Workloads

Francisco Webber
CEO
Cortical.io

September 2022

# AI EFFICIENCY CHALLENGES

## Energy Efficiency

- Currently the principal way to improve the accuracy of natural language ML-models is to apply brute-force by increasing the number of neurons/neuron layers/parameters always leading to more computations. As mostly dot products are calculated, adding a single feature can double the number of multiplications necessary.

Google: *"One popular model, GPT-3, has 175 billion machine learning parameters. It was trained on NVIDIA V100, but researchers have calculated that using A100s would have taken 1,024 GPUs, 34 days and $4.6million to train the model. While energy usage has not been disclosed, it's estimated that GPT-3 consumed 936 MWh"*

- The global energy consumption of the digital domain is estimated to be equivalent to the level of the global air traffic, matching the global automobile traffic in 2030.

## Data Efficiency

- Huge data collections must be applied to train general purpose language models, needed to be applied to the variety of necessary use cases.
- State-of-the-art language models are currently available in English only as for most other languages there does not even exist enough raw material.

## Model efficiency

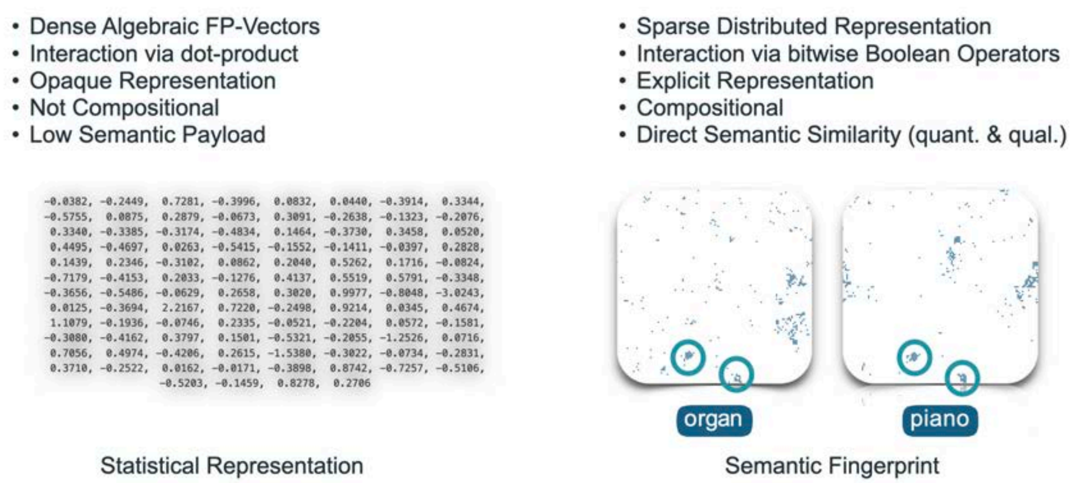
- Every specific use case domain requires its own independent model (e.g. "Complaint Email Classification in tele-marketed household appliances", "Support Ticket Attribution for Network Technology Providers" etc.).
- Models need to be very specific to deliver useful performance in the business context, in most cases the amounts of data needed for training cannot be provided.

# A THREE PART INNOVATION APPROACH

However, High Efficiency AI can be achieved by leveraging three innovations: (1) Semantic Folding, a new kind of data representation; (2) the Proximus development tool; and (3) Hardware acceleration with Xilinx FPGA and AMD EPYC Milan

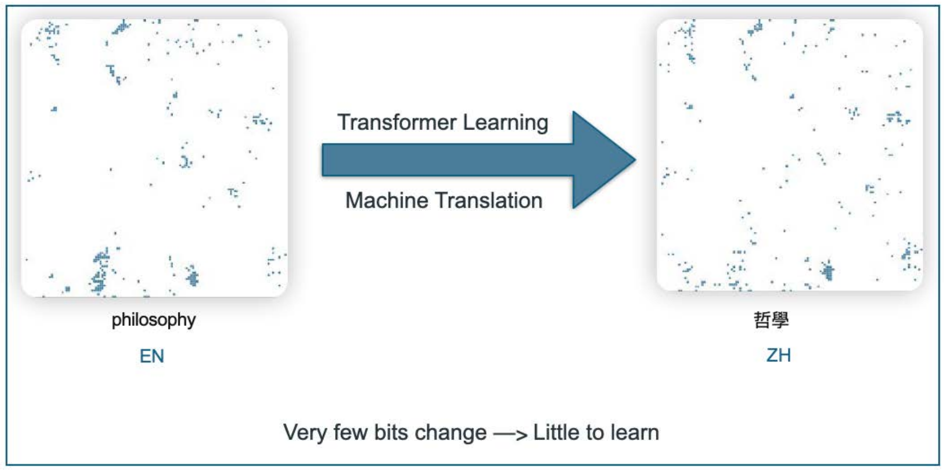
## A new kind of data representation: Semantic Folding

- The inevitable combinatorial explosion that occurs with fully connected ANNs, where feature vectors must be processed as endless series of matrix dot products, is replaced by explicit and independent features during the Semantic Folding process. This enables the efficient application of massive parallelization schemes such as aggregate/de-aggregate and systolic pipelining during learning and inference phases, which improves the training speed by several orders of magnitude.

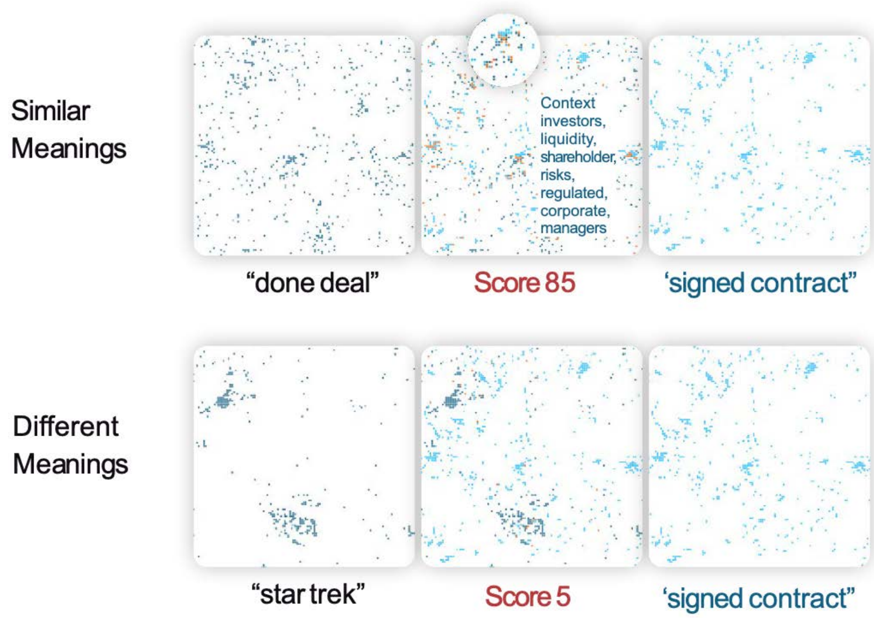



- The Semantic Folding process also increases the semantic payload of the representations by attributing topology information (geometrical context) to each basic feature. This is why DNNs work particularly well for image data, since two adjacent pixels that share a similar color can be assumed to originate from the same "object" captured in the image, with the object being the actual semantic content of the image.

Neural Machine Translation from English into Chinese

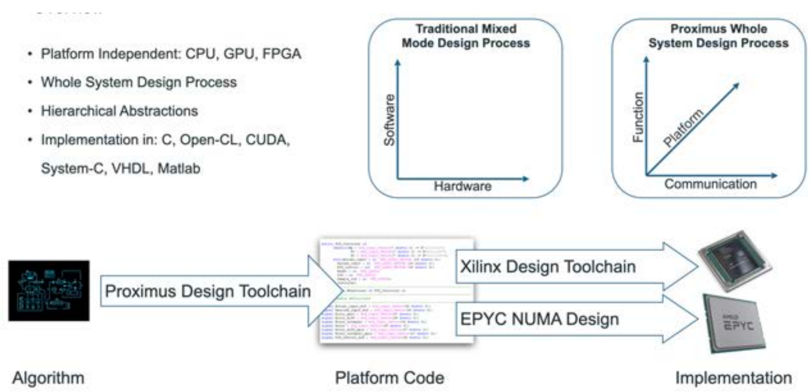



- The complete locality of the Semantic Folding type of features reduces the similarity operator- fundamental to all machine learning processes- to bit-wise Boolean operations on binary vectors. This leads to very efficient inference calculations and enables very high sustained data throughput when applying models to production workloads.

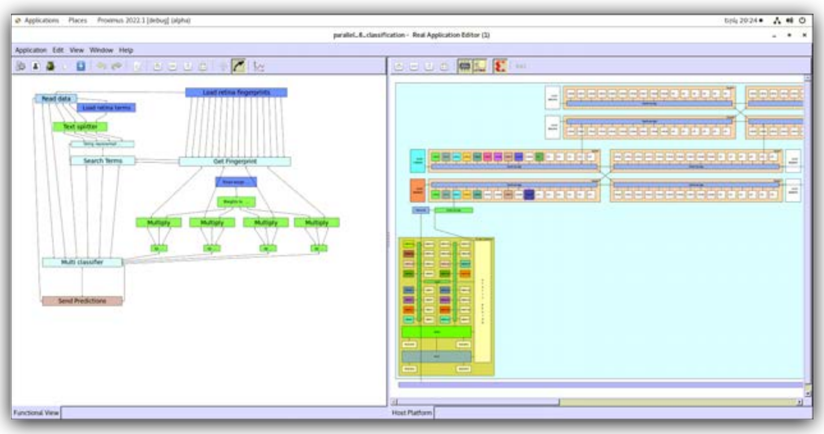


## Proximus enables a fully integrated design flow

- By integrating Proximus into the design process, Cortical.io had the opportunity to freely choose between a software-bound, processor-bound, or memory-bound implementation for each individual computational step, making it easier to approach the theoretical maximum performance of the algorithm as a whole.



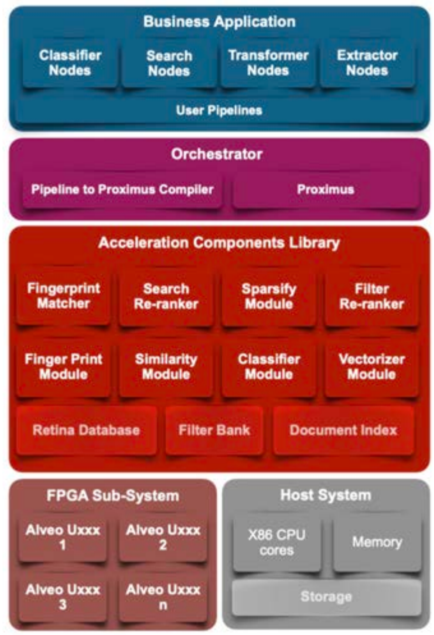

- Proximus enables switching the software/hardware domain forth and back within the same engineering/design process, which has historically proven to be a major obstacle when developing computation-task-specific silicon.



- Proximus allows the “compilation” of each system design for many different processing platforms such as AMD, Intel, Arm, RISC-V processors, GPU, FPGA to ASIC implementations using all types of interconnect fabrics and architectures as well as memory technologies from static RAM cells over DDR to HBM.

## Specific accelerator elements on Silicon for Xilinx FPGA and AMD EPYC Milan

- Functional Modules: Fingerprint Encoder, Tokenizer / Parser, Vectorizer, Classifier, Fingerprint Search
- Design Patterns: Aggregator, De-Aggregator, HBM Streamer, FP Matcher, B-RAM Searcher
- Example of an FPGA-specific advantage: Using on-chip B-RAM memory as CAM (Content Addressable Memory) allows a one-step access to the precomputed fingerprint dictionary. Converting a token into a lossless hash directly generates the B-RAM location on the chip, where the address of the associated fingerprint is stored. The actual fingerprint values are stored in Alveo's HBM memory, which is attached via 32 independent memory channels.




- Example of an AMD EPIC Milan-specific feature: By leveraging the processor-specific NUMA architecture/functionalities and the local (non-interdependent) nature of the fingerprint features, data can be optimally segmented horizontally or vertically depending on the given processing step. This ensures that each processor core only needs to access memory locations within its NUMA segment, whereby all available cores are equally utilized to their maximum theoretical throughput.

# BENCHMARK RESULTS

- Specification of the used host system
  - Dual AMD EPYC Milan processors with 64 cores each
  - 256GB DDR4 RAM
  - Either no expansion cards/ 2 GPU cards / 4 Xilinx ALVEO U50 cards
- Evaluation datasets
  - 16 individual public datasets for the document classification task
  - The collections contain between 2225 to 630K documents
  - Document length varies from 15 to 390 terms

| Input file | Documents | Document Length (Terms) |
|---|---|---|
| Jeopardy (20) | 6874 | 15 |
| DBpedia | 630000 | 46 |
| Jeopardy (10) | 3531 | 15 |
| SemEval-2017 Task 4 | 32538 | 18 |
| PubMed | 240387 | 26 |
| Reuters R8 | 7692 | 104 |
| Enron (Kaminski) | 4255 | 237 |
| BBC News Text | 2225 | 390 |
| Web of Science Fine-Grained | 46946 | 200 |
| 20 Newsgroups | 18793 | 258 |
| Enron (Farmer) | 3553 | 135 |
| Enron (Lokay) | 2334 | 233 |
| Web of Science | 46985 | 200 |
| Reuters R52 | 46985 | 112 |
| Ohsumed | 56984 | 166 |
| IMDb | 50000 | 231 |

Energy consumption of the different configurations have been averaged to:

| | |
|---|---|
| BERT-GPU | 1500 Wh |
| Java Enterprise | 1000 Wh |
| EPYC-Milan | 1000 Wh |
| 4x ALVEO U50+ EPYC-Milan | 1300 Wh |

| | Bert (GPU) | Semantic Folding | Semantic Folding | Semantic Folding |
|---|---|---|---|---|
| | Python Classifier | Enterprise Java Classifier | AMD Epyc Milan Classifier | 4Card FPGA + AMD Epyc Milan Classifier |
| MB/sec per 1U Rack Space | 0,18 | 18,42 | 180,3 | 528,30 |
| Acceleration | 1x | 100x | 1000x | 2865x |
| Energy per MB | 2.26 Wh | 15 mWh | 1,54 mWh | 0,46 mWh |
| Efficiency Factor | 1x | 150x | 1467x | 4913x |

**Acceleration: x2800**

**Energy Efficiency: x4900**

**Compared to the State-of-the Art**

cortical.io
Vienna | New York | San Francisco
info@cortical.io | sales@cortical.io | www.cortical.io

September 2022