ME

- Junior Scientist at TNO in The Hague
  - Department Applied Cryptography and QUantum Applications
  - Main focus: Quantum Applications
  - Talk: Graphs (and Applied Cryptography)

- Background
  - Studied mathematics at the University of Cambridge
  - Masters: specialised in combinatorics

- Live in Breda
  - Love climbing, hockey and spikeball

Dr. ir. P.N. Langenkamp
PROJECT

- Alliance for Privacy-Preserving Detection of Financial Crime
  - Consortium of banks, CWI and TNO
  - Different subprojects

- Goal: aid banks in detecting money laundering while preserving privacy
  - Money laundering: The process of concealing the origin of money, often obtained from illicit activities such as drug trafficking, corruption, embezzlement or gambling, by converting it into a legitimate source
  - My subproject: graph embeddings

GRAPH EMBEDDINGS

GOAL

- Use machine learning to aid the detection of money laundering
  - Identify money laundering patterns
  - Identify individuals involved in money laundering

- Input for machine learning model:
  - Transaction graph with nodes = accounts, edges = transactions
  - Each node has certain features/attributes, such as cashflow, sort of account etc.

- Problems:
  - How do we use machine learning on large transaction graphs?
  - What if part of the graph is out of our scope?
**WHAT ARE GRAPH EMBEDDINGS?**

“Graph embedding is an approach that is used to transform nodes, edges, and their features into vector space (a lower dimension) *whilst maximally preserving properties like graph structure and information*. Graphs are tricky because they can vary in terms of their scale, specificity, and subject.”

Flawnson Tong; Graph Embedding for Deep Learning; towardsdatascience.com; May 6, 2019

- Graph embedding techniques
  - Take a graph
  - Embed in lower dimensional space
    - Node-level, (sub)graph-level or through strategies like graph walks
  - Pass on to machine learning model
    - e.g. Random forest classifier

**HOW TO MAKE A GRAPH EMBEDDING**

**DIFFERENT APPROACHES (TWO EXAMPLES)**

**DEEPWALK (PEROZZI ET AL.)**

- Uses local information obtained from truncated random walks to learn latent representations by treating walks as the equivalent of sentences
- Does not do well at preserving the local neighborhood of nodes

**GRAPHSAGE (HAMILTON ET AL.)**

- Inductive representation learning on large graphs
- Especially useful for graphs that have rich attribute information
- **GraphSAGE**

**Algorithm**

- Samples neighbourhood, Aggregating sampled information
- Works iteratively per layer
- GraphSAGE only considers node labels, no edge labels (e.g., transaction amounts)

**Overview Picture**
1. **Sampling**  
   For each node, sample neighbours at random

2. **Aggregation**  
   For each node, aggregate the node labels of the sampled children

3. **One-layer neural network**  
   Use activation function $\sigma$, e.g. identity, and some weight matrix (either to be trained or already trained)

   
   $$h^k_b = \sigma(W_k(\text{aggregate sampled neighbours } s(h^{k-1}_b, h^{k-1}_u)))$$

   Repeat 2 and 3 one or two times

---

**GRAPH EMBEDDING – HOW TO DETERMINE WEIGHTS**

**GRAPHSAGE**

**TWO PHASES**

- Training phase; to determine weights
- Run a basic machine learning model
- Different weights for each layer
- Implementation phase; use weights from training to generate graph embeddings
ADDED VALUE GRAPH EMBEDDINGS
CORA DATASET AS AN EXAMPLE

- Cora dataset:
  - Nodes: scientific publications (2708), classified into one of seven classes.
  - (Directed) Edges: citations (5429)
  - Features: 0/1-value whether a citation contains a certain word (1433 unique words)

- Goal: classify papers correctly in their class

- Assumption: neighboring nodes give information about the class of this node.
  - This makes GraphSAGE worth while, since we embed information of the neighboring nodes into the graph embedding of the nodes.

GRAPH EMBEDDINGS VS. INITIAL FEATURES

- In order to say something useful about using graph embeddings (resulting in 50 features) for the Cora Dataset, we also want to compare with using the initial features (1433 features) without using the network structure.

Machine learning model on original features: accuracy of 0.584
Machine learning model after GraphSAGE: accuracy of 0.783

Conclusion: Using GraphSage for the Cora Dataset has a clear added value.
HOW TO APPLY GRAPHSAGE IN FINANCE

To apply GraphSAGE, require node attributes from neighbours

- In general, this will include accounts at other banks. Banks cannot share
  this information freely → need cooperation in a secure way

Solution: cryptography!
- Study of secure communication techniques
- Uses encryption and decryption

Interesting for us: Secure Multi-Party Computation (MPC)
- Doing computations on shared data without revealing the data
- Only reveals the result

APPLY MPC FOR GRAPHSAGE

IDEA

- Downside: working in the encrypted domain gives large computational and communicational overheads.

- Training phase already takes 6 hours without using encryption. Unfeasible to do in the encrypted domain.
  Instead, calculate weight matrix together in a smart way.
  - Run training phase locally
  - Aggregate intermediate weight matrices after each epoch

- Implementation phase can be made securely with an easy application of MPC.
  - Reason: all operations are additions or multiplications
**DISTRIBUTED TRAINING PHASE**

- train_A
- train_B
- train_C

\[
\text{train}_\text{all} = \text{train}_\text{A} + \text{train}_\text{B} + \text{train}_\text{C}
\]

**SECURE IMPLEMENTATION PHASE**

- test_A
- test_B
- test_C

\[
\text{test}_\text{all} = \text{test}_\text{A} + \text{test}_\text{B} + \text{test}_\text{C}
\]

Weight matrix

\[
W_A
\]

Take mean after each epoch

\[
W_B
\]

Aggregate

\[
W_C
\]

\[
W_X
\]

Dr. ir. P.N. Langenkamp
GraphSAGE
- Promising algorithm to aid detection of money laundering
- Can be applied in a secure way using MPC

Main results
- Wrote a secure implementation of GraphSAGE in python
- Successful tests on different examples

Next steps
- Write a demo to show the workings of GraphSAGE to the banks
- Turn it into a deliverable so banks can start using it on real data
EXAMPLE OF CRYPTOGRAPHY
RSA

- Take (large) primes $p$ and $q$ and compute $N = p \times q$.
- Knowing the primes, it is easy to compute $\varphi(N) = \varphi(p) \times \varphi(q) = (p - 1)q - 1$. Hard to compute knowing just $N$.
- Choose $1 < e < \varphi(N)$ coprime to $\varphi(N)$.
- Publish $(e, N) =$ public key. Note: hard to compute $p$ and $q$ knowing only $N$.
- Compute inverse $d$ of $e \mod \varphi(N)$ and keep as private key.

- Encrypt message $a < N$ by computing $[a] = a^e \mod N$.
- Decrypt $c$ by computing $c^d = a^{ed} = a \mod N$.
- Mathematical theorem: given $a^e$, hard to compute $a$ without knowing factorisation of $N$.

- Homomorphic property: preserve mathematical operations such as $+$ and $\times$ while encrypting
  - $[a + b] = (a + b)^e = a^e + b^e = [a] + [b]$