Feature Engineering for Causal and Predictive Inference

Alexander Quispe
The World Bank, PUCP
aw.quispero@up.edu.pe

June 18, 2024

Table of Contents

Introduction

From Principal Components to Autoencoders

From Auto-Encoders to General Embeddings

Text Embeddings
- First Generation: Word2Vec Embeddings
- Second Generation: Sequence Models
- Third generation: Transformers

Image Embeddings

Applicacion: Hedonic Prices

Introduction

Complex Data for Inference

We have previously restricted our inference to simple numeric datasets. However, researchers now often deal with complex data like images and text. We extend our approach to include these data types.

- Example: Predicting product prices using text descriptions and images.
- Predicted prices, known as hedonic prices, are based on hedonic price models from economics.

Creating Embeddings for Price Prediction

To predict prices, we convert text and images into low-dimensional numerical features, called ”embeddings.”

- Minimum requirement: Similar products have similar embeddings.
- Maximum requirement: Embeddings capture maximal information relevant to price predictions.
- Methods:
  - Classical principal component analysis
  - Auto-encoders
  - Neural networks for auxiliary tasks (e.g., object classification, image compression, text summarization)

Embeddings in Neural Networks

Neural networks like BERT are trained on auxiliary tasks to learn word similarity and contextual meaning.

- BERT creates embeddings summarizing product text descriptions.
- These embeddings are useful for price prediction, as descriptions influence prices.
- Embeddings can be used in various predictive and causal inference problems, such as:
    - Modeling product variety and demand.
    - Studying wage offer structures with text resumes.
    - Analyzing the effect of institutions using country characteristics.

From Principal Components to Autoencoders

Principal Component Analysis (PCA)

Principal components are classical examples of embeddings. PCA finds orthogonal linear combinations of variables that best reproduce the data. A small number of principal components capture most of the data's variability, providing a useful low-dimensional summary.

- Let $(W_1, ..., W_n)$ be a sample of n observations of a high-dimensional centered random vector $W_i \in \mathbb{R}^d$.
- Empirical covariance matrix: $\Sigma_n = \mathbb{E}_n[WW'] \in \mathbb{R}^{d \times d}$.

Principal Component Analysis (PCA)

- Reduce the dimension by finding $K \ll d$ orthogonal rotations:

$$X_{ki} := c_k' W_i, \quad k = 1, ..., K,$$

  where $c_k' c_\ell = 0$ for $\ell \neq k$ and $c_k' c_k = 1$ for each k.
- These rotations X_{ki} are the principal components of W_i.
- Objective: Solve

$$\min_{\{a_j\}_{j=1}^d, \{c_k\}_{k=1}^K} \sum_{j=1}^d \sum_{i=1}^n (W_{ji} - \hat{W}_{ji})^2,$$

  subject to

$$\hat{W}_{ji} := a_j' X_{Ki}, \quad X_{Ki} = (X_{i1}, ..., X_{iK})',$$

$$X_{ki} = c_k' W_i, \quad c_k' c_k = 1, \quad c_k' c_\ell = 0 \text{ for } \ell \neq k.$$

- The constructed variables $X_{Ki} = (X_{1i}, ..., X_{Ki})'$ are the first K principal components.

Properties of Principal Components - Linear Encoders

Principal components satisfy important properties due to the eigenvectors c_k of Σ_n.

- $\mathbb{E}_n[X_k^2] = \lambda_k$ for $k = 1, \dots, K$
- $\mathbb{E}_n[X_k X_\ell] = 0$ for $\ell \neq k$

These properties enable the use of principal components for encoding/embedding raw inputs. Similarity between raw inputs W_k and W_ℓ can be measured by cosine similarity of their embeddings:

$$\text{sim}(W_k, W_\ell) = \frac{X_k' X_\ell}{\|X_k\| \|X_\ell\|}$$

A Linear Single Layer Auto-encoder

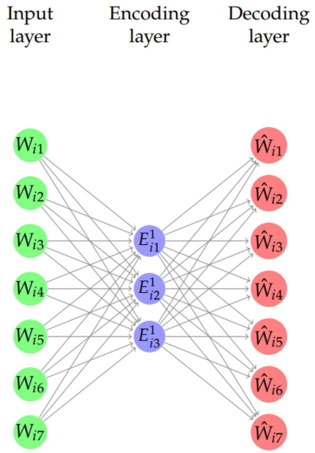


Figure: A linear single layer autoencoder

Properties of Principal Components - NonLinear Encoders

In nonlinear encoders and decoders with multiple layers:

$$W_i \xrightarrow{g_1} E_{1i} \xrightarrow{g_2} \cdots \xrightarrow{g_k} E_{ki} \xrightarrow{g_{k+1}} D_{k+1i} \xrightarrow{g_m} D_{mi} = \hat{W}_i,$$

where g_ℓ are neuron-generating maps. The middle layers E_{ki} represent low-dimensional encodings.
These well-behaved encodings generalize PCA to nonlinear methods, enhancing the ability to find similarities in complex data.

Three Layer Non-Linear Auto-encoder

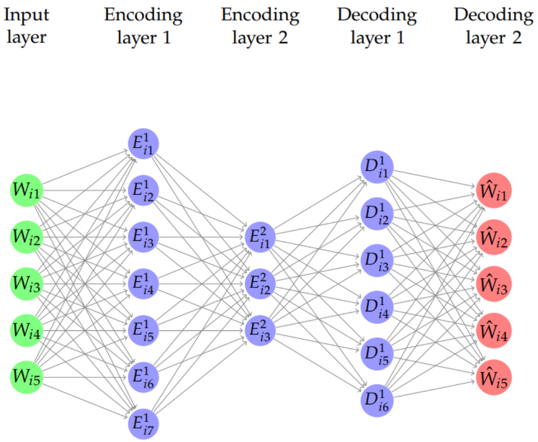


Figure: Three layer nonlinear autoencoder

From Auto-Encoders to General Embeddings

From Auto-Encoders to General Embeddings (1/2)

The concept of loss extends to other functions where the target outcome A may not be W, but something else. Embeddings are sought to minimize average prediction loss:

$$\mathbb{E}_n[\text{loss}(A, f(e(W))],$$

where $f(\cdot)$ predicts A.

From Auto-Encoders to General Embeddings (2/2)

For instance, in image feature engineering, A could be a product type and W the image. In text feature engineering, A could be a masked word in a sentence and W the sentence.
These approaches may be more relevant to the final learning task.
Implemented via neural networks:

$$W_i \xrightarrow{g_1} E_{1i} \xrightarrow{g_2} \cdots \xrightarrow{g_k} E_{ki} \xrightarrow{g_{k+1}} F_{k+1i} \xrightarrow{g_m} F_{mi} =: \hat{A}_i,$$

where g_ℓ are neuron-generating maps. E_{ki} are embedding layers, and F_{ki} are predictive layers for auxiliary targets.

Text Embeddings

First Generation: Word2Vec Embeddings

Word2Vec Embeddings (1/3)

Word2Vec, introduced by Mikolov et al., aims to encode words into lower-dimensional vectors while capturing word similarity.

- High-dimensional encodings like one-hot vectors lack utility and fail to capture word similarity.
- Word embeddings are represented by a reduced-dimensional matrix $\Omega = \{u_1, \ldots, u_d\}$, derived from a linear rotation of the original dictionary.
- The objective is to find an effective representation with dimension much smaller than the total number of words in the corpus.
- Word similarity is captured through the cosine similarity of embeddings.

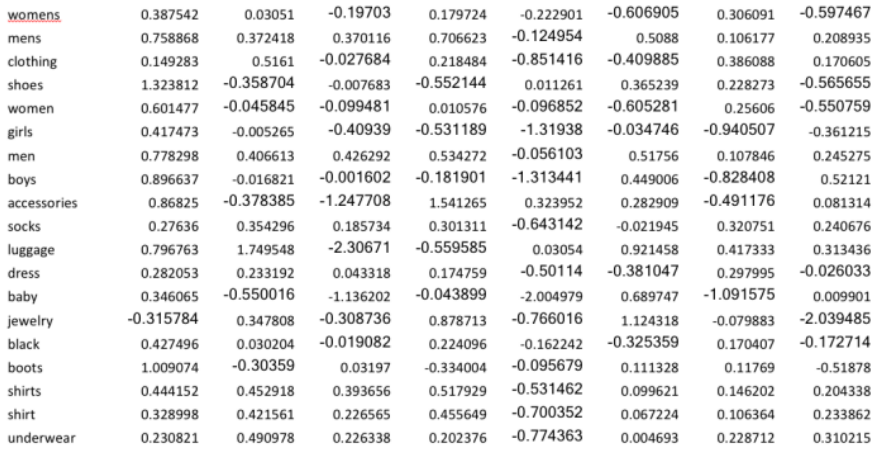
Word2Vec Embeddings (2/3)

To predict a central word given context words, Word2Vec uses a multinomial logit function:

$$p_s(t; \pi, \Omega) = \frac{exp(\pi_t' \bar{U}_s(\Omega))}{\sum_{\bar{t}} exp(\pi_{\bar{t}}' \bar{U}_s(\Omega))}$$

- π is a matrix of parameter vectors defining choice probabilities.
- Context words are collapsed by summing their embeddings.
- The choice probabilities are constrained to Ω and estimated using maximum quasi-likelihood method.

Word2Vec Embeddings (2/3) - Embebdding Table Example



womens	0.387542	0.03051	-0.19703	0.179724	-0.222901	-0.606905	0.306091	-0.597467
mens	0.758868	0.372418	0.370116	0.706623	-0.124954	0.5088	0.106177	0.208935
clothing	0.149283	0.5161	-0.027684	0.218484	-0.851416	-0.409885	0.386088	0.170605
shoes	1.323812	-0.358704	-0.007683	-0.552144	0.011261	0.365239	0.228273	-0.565655
women	0.601477	-0.045845	-0.099481	0.010576	-0.096852	-0.605281	0.25606	-0.550759
girls	0.417473	-0.005265	-0.40939	-0.531189	-1.31938	-0.034746	-0.940507	-0.361215
men	0.778298	0.406613	0.426292	0.534272	-0.056103	0.51756	0.107846	0.245275
boys	0.896637	-0.016821	-0.001602	-0.181901	-1.313441	0.449006	-0.828408	0.52121
accessories	0.86825	-0.378385	-1.247708	1.541265	0.323952	0.282909	-0.491176	0.081314
socks	0.27636	0.354296	0.185734	0.301311	-0.643142	-0.021945	0.320751	0.240676
luggage	0.796763	1.749548	-2.30671	-0.559585	0.03054	0.921458	0.417333	0.313436
dress	0.282053	0.233192	0.043318	0.174759	-0.50114	-0.381047	0.297995	-0.026033
baby	0.346065	-0.550016	-1.136202	-0.043899	-2.004979	0.689747	-1.091575	0.009901
jewelry	-0.315784	0.347808	-0.308736	0.878713	-0.766016	1.124318	-0.079883	-2.039485
black	0.427496	0.030204	-0.019082	0.224096	-0.162242	-0.325359	0.170407	-0.172714
boots	1.009074	-0.30359	0.03197	-0.334004	-0.095679	0.111328	0.11769	-0.51878
shirts	0.444152	0.452918	0.393656	0.517929	-0.531462	0.099621	0.146202	0.204338
shirt	0.328998	0.421561	0.226565	0.455649	-0.700352	0.067224	0.106364	0.233862
underwear	0.230821	0.490978	0.226338	0.202376	-0.774363	0.004693	0.228712	0.310215

Figure: Word2Vec: Word Embeddings for ’shirt’ vs ’shirts’ and ’luggage’ vs ’dress’

Word2Vec Embeddings (3/3)

Evaluation of Word2Vec embeddings involves:

- Assessing improvement in prediction quality by Word2Vec features.
- Measuring similarity between words using cosine similarity of embeddings.
- Word embeddings enable analogical reasoning, demonstrated by the construction of artificial latent words.

Word2Vec was among the early successful embedding algorithms, paving the way for advanced NLP techniques like ELMo and BERT.

Second Generation: Sequence Models

Sequence Models

ELMo (Embeddings from Language Models) utilizes recurrent neural networks to represent text sequences, improving context understanding.

Forward Prediction:

$$p_{f_{k,m}}(t) = P[T_{k+1,m} = t | T_{1,m}, \ldots, T_{k,m}; \theta]$$

Backward Prediction:

$$p_{b_{k,m}}(t) = P[T_{k-1,m} = t | T_{k,m}, \ldots, T_{n,m}; \theta]$$

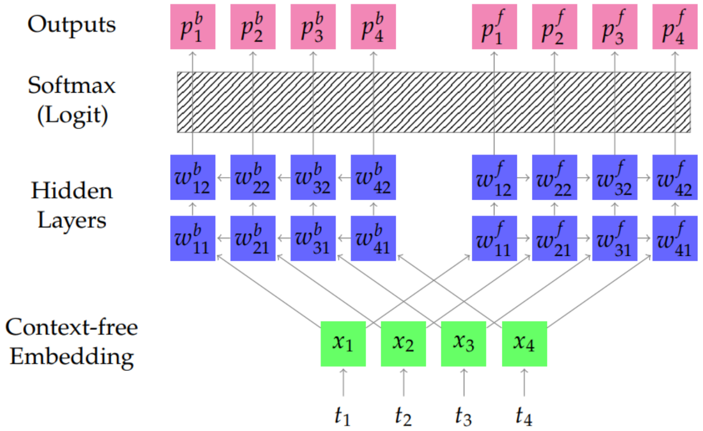
- RNNs model forward and backward probabilities.
- Parameters are estimated by maximizing quasi-log-likelihoods.

ELMo: Structure

ELMo employs a recursive nonlinear regression model to build probabilities.
Modeling Forward Prediction:

$$P(T_{k,m} = t | \{T_{j,m}\}_{j=1}^{k-1}) = \frac{\exp(\sum_{j=1}^{k-1} \pi'_{t,j} U_{j,m}(\omega))}{\sum_{\bar{t}} \exp(\sum_{j=1}^{k-1} \pi'_{\bar{t},j} U_{j,m}(\omega)')}$$

ELMo Architecture



Outputs
Softmax (Logit)
Hidden Layers
Context-free Embedding
p_1^b p_2^b p_3^b p_4^b
p_1^f p_2^f p_3^f p_4^f
w_{12}^b w_{22}^b w_{32}^b w_{42}^b
w_{12}^f w_{22}^f w_{32}^f w_{42}^f
w_{11}^b w_{21}^b w_{31}^b w_{41}^b
w_{11}^f w_{21}^f w_{31}^f w_{41}^f
x_1 x_2 x_3 x_4
t_1 t_2 t_3 t_4

Figure: ELMo network for a string of 4 words, with $L = 2$ hidden layers.

ELMo: Training

Training involves maximizing the quasi-log-likelihoods of observed data.

$$\max_{\theta} \sum_{m \in M} \sum_{k=1}^{n-1} \log p_{f_{k,m}}(T_{k+1,m}; \theta) + \sum_{k=2}^{n} \log p_{b_{k,m}}(T_{k-1,m}; \theta)$$

where M is the collection of sentences, and $p_{f_{k,m}}$ and $p_{b_{k,m}}$ are forward and backward prediction probabilities.

ELMo: Producing Embeddings

To produce embeddings, each word t_k in a sentence $m = (t_1, ..., t_n)$ is mapped to a weighted average of the outputs of hidden neurons indexed by k:

$$t_k \mapsto w_k := \sum_{i=1}^{L} (\gamma_i w_{ki}^f +_i w_{ki}^b)$$

- Each word t_k is mapped to a weighted average of outputs of hidden neurons.
- Embedding for a sentence is produced by summing embeddings for individual words.
- Weights γ can be tuned by the neural network.

Third generation: Transformers

Transformer Architecture: BERT

A subsequent major advance in language modeling has been the development and use of the transformer architecture.
Transformers utilize ”self-attention” mechanism to model the importance of different parts of the text in understanding each other, enabling better context comprehension compared to RNNs.
Bidirectional Encoder Representations from Transformers (BERT). Unlike ELMo, BERT is trained on two self-supervised tasks:

- Masked Language Model: Randomly mask words in a sentence and predict them.
- Next Sentence Prediction: Predict whether one sentence precedes another.

BERT: Structure

1. Tokenization: Each word is broken into subwords (tokenized) and encoded using WordPiece. Special tokens like [cls] and [mask] are added.
2. Input Representation: Each token's input representation consists of token embedding, position embedding, and segment embedding.
3. Main Model Architecture: Input representations are fed into Transformer-Encoder blocks, consisting of multi-head attention layers and feedforward layers.
4. Output and Loss: Output representations are used for masked word prediction and next sentence prediction, with loss function combining both losses.

BERT significantly advances language modeling capabilities, especially with its "multi-head attention" layer.

BERT: Computing the Attention (1)

- Begin with context-free embeddings $(x_1, x_2, \ldots, x_n)$, where each $x_k \in \mathbb{R}^d$.
- Transform each token embedding x_k into a key embedding κ_k and a value embedding v_k.
- Use a query vector q to calculate the similarity between the query and key vectors, resulting in a selection probability.
- The embedding of the neighborhood corresponding to the query is the weighted average of the value embeddings.

In matrix form, the attention embeddings can be written as:

$$A = \text{Attention}(Q, K, V) = \sigma\left(\frac{QK^\top}{\sqrt{d_k}}\right) V$$

BERT: Computing the Attention (2)

- Given n neighborhood queries $q_1, \ldots, q_n$, we create corresponding neighborhood embeddings $a_1, \ldots, a_n$.
- Each Transformer building block in BERT comprises several repetitions of multi-head attention encodings, followed by a fully connected neural network applied to each of the n output encodings separately.

Overall, this transformation takes as input a matrix $X \in \mathbb{R}^{n \times d}$ and transforms it into a matrix A, where each row k corresponds to the neighborhood embedding associated with query q_k, which in turn is associated with token x_k.

BERT: Generating Product Embeddings

- Depending on specific tasks and resources
    - Use the last layer, second-to-last layer, or concatenate the last 4 layers of the encoder outputs from the pre-trained BERT model.
    - Fine-tune the whole BERT model using the downstream task.
    - Train the BERT language model from scratch on new data.

BERT: Architecture

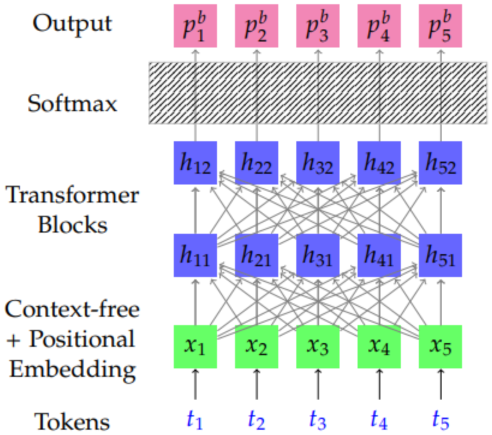


Figure: Bidirectional Encoder Representations from Transformers (BERT) Architecture

Image Embeddings

Residual Networks (ResNet50)

ResNet50, developed by He et al., is a successful deep learning model for image classification, particularly for ImageNet and COCO datasets.

- ResNet50 exploits ”partial linearity” by adding skip connections to traditional feed-forward convolutional neural networks.
- Each residual block predicts a residual:

$$v \xrightarrow{\omega_{0k}} (v, \sigma_{0k}(\omega_{0k} v)) \xrightarrow{\omega_{1k}} (v, \sigma_{1k} \circ \omega_{1k}\sigma_{0k}(\omega_{0k} v)) \to v + \sigma_{1k} \circ \omega_{1k}\sigma_{0k}(\omega_{0k} v)$$

- The architecture addresses vanishing gradients by directly including the residual via skip connections, allowing for training of very deep networks.
- The last hidden layer serves as the image embedding.

ResNet50

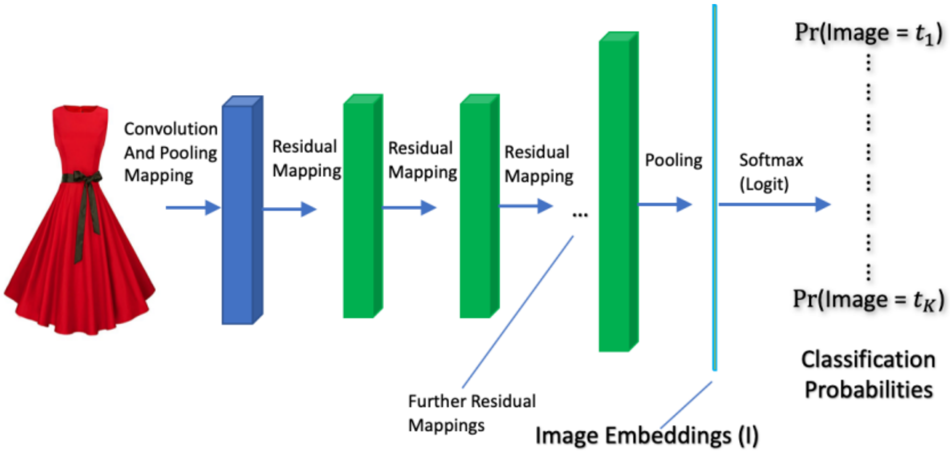

Convolution And Pooling Mapping
Residual Mapping
Residual Mapping
Residual Mapping
...
Pooling
Softmax (Logit)
$\Pr(\text{Image} = t_1)$
$\Pr(\text{Image} = t_K)$
Classification Probabilities
Further Residual Mappings
Image Embeddings (I)

Figure: Operation of ResNet50 on Three-Dimensional Image Arrays

Applicacion: Hedonic Prices

Hedonic Price Models

- Empirical hedonic model: Predicts price based on product characteristics.
- Prediction equation:

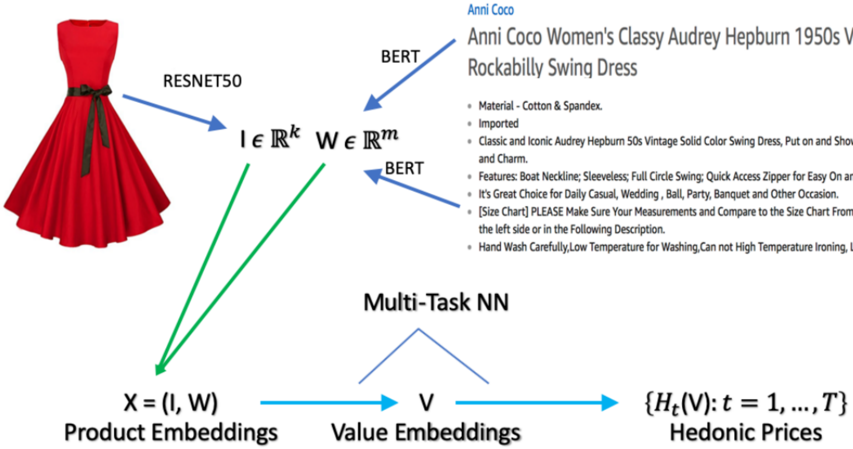
$$P_{it} = H_{it} + \epsilon_{it} = h_t(X_{it}) + \epsilon_{it}, \quad E[\epsilon_{it} | X_{it}] = 0$$

- Where:
  - P_{it}: Price of product i at time t
  - X_{it}: Product features
  - $h_t(\cdot)$: Price function
  - ϵ_{it}: Error term

Structure of the Predictive Model

- **Data Preprocessing:**
  - Input: Images and unstructured text.
  - Transformation: Embedding data into numerical vectors I and W using state-of-the-art deep learning techniques such as ResNet50 and BERT.
- **Prediction of Hedonic Prices:**
  - Input: $X_{it} = (W_{it}', I_{it}')'$
  - Method: Utilization of deep neural networks with a multitask structure.
  - Process:
    - Training of models on tasks unrelated to price prediction.
    - Extraction of embeddings as hidden layers of the neural networks.
    - Training of models on price prediction tasks.
  - Creation of an intermediate lower dimensional embedding $V = V(X)$.
  - Prediction of final prices in all time periods $\{H_t(V), t = 1, ..., T\}$.

Model Representation


Anni Coco

Anni Coco Women's Classy Audrey Hepburn 1950s Vintage Rockabilly Swing Dress

- Material - Cotton & Spandex.
- Imported
- Classic and Iconic Audrey Hepburn 50s Vintage Solid Color Swing Dress, Put on and Show Your Elegance and Charm.
- Features: Boat Neckline; Sleeveless; Full Circle Swing; Quick Access Zipper for Easy On and Off
- It's Great Choice for Daily Casual, Wedding , Ball, Party, Banquet and Other Occasion.
- [Size Chart] PLEASE Make Sure Your Measurements and Compare to the Size Chart From the picture on the left side or in the Following Description.
- Hand Wash Carefully,Low Temperature for Washing,Can not High Temperature Ironing, Line Dry

Figure: Structure of the Predictive Model