Forecasting Spatio-Temporal Data with Bayesian Neural Networks

Tesi di Laurea Magistrale di Federico Ravenda

Relatore: Stefano Peluso
Correlatore: Mirko Cesarini

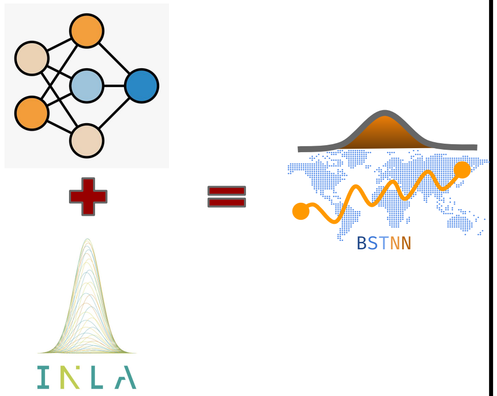
Research Question & Goal

- Can a Neural Network imitate traditional statistical Spatio-Temporal models?
- How to model the spatial and temporal components?

Why to do that?

To exploit both the advantages of neural networks:
- Flexibility
- No strict statistical assumptions

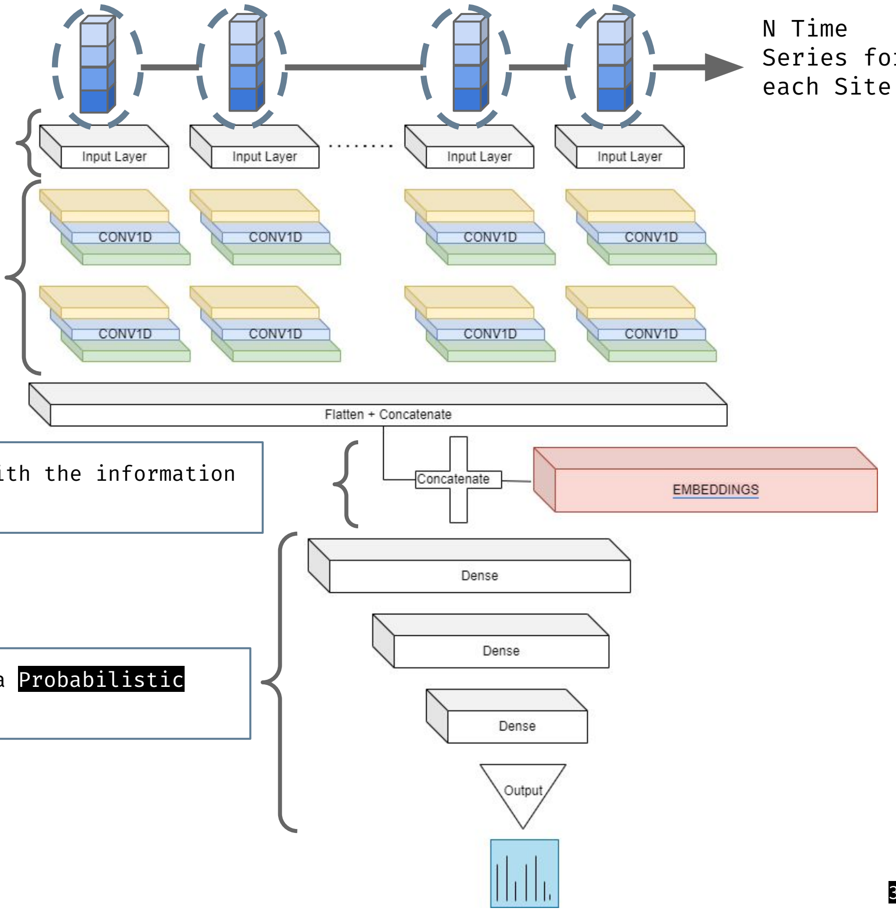
and the advantage of Bayesian Hierarchical Models:
- To account and quantify the uncertainty



Main Intuition: **Transpose** relevant objects from a Bayesian Hierarchical Model, **INLA**, into a Machine Learning Framework → *Bayesian Spatio-Temporal Neural Network* (**BSTNN**)

Bayesian Spatio-Temporal Neural Network (BSTNN)

Considering N different sites and T temporal instants. We want to make predictions for the following T + h temporal instants.



There are as many Input Layer as the number of Locations N.

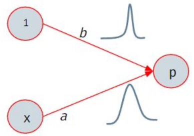
1-Dimensional Convolutional Layers provide a nice architecture to learn smoothing parameters.

Different types of Embeddings (spatial and temporal) are concatenated with the information returned by Conv1D Layers.

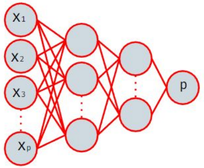
Dense Layers are stacked until the output Layer where a Dense Layer or a Probabilistic Layer can be used.

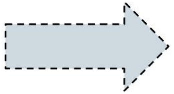
...But Why "BAYESIAN" 

Bayesian Linear Regression

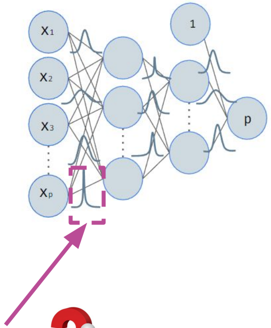


Neural Network





Bayesian Neural Network



They can take into account to 2 different types of Uncertainty

Aleatoric Uncertainty

Uncertainty in Data. In Statistical Modelling, it is often accounted for by introducing probabilistic models that capture the inherent noise in the data.

Epistemic Uncertainty

It reflects the uncertainty in the model itself. In Statistical Modelling, it is often accounted for through the use of probabilistic models, which allow for uncertainty in the model parameters.

How  Using Approximation Methods

MCMC

- It is used to approximate complex distributions that are difficult to sample from directly.
- It works for small problems,say 10 to 100 variables.

Variational Inference

- Each weight is replaced by a distribution.
- Complicated Posterior Distribution of the weights are approximated by a simple distribution called VARIATIONAL DISTRIBUTION.

MC Dropout

- It is a Bayesian extension of Dropout.
- Predictions are based on MC samples.

Different Types of Embeddings...

Embeddings are a **low dimensional** space into which we translate **high-dimensional vectors**, to model the spatial (and temporal) components in order to feed them to neural networks.

Static Spatial Component

Use of Node2Vec to extract Geographical information.

Static Spatial & Temporal Components

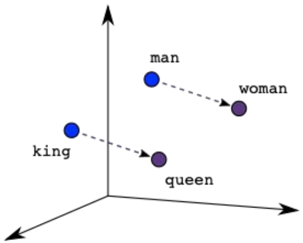
Use of Entity Embedding to extract intrinsic location characteristics and Temporal information i.e., Months, years, etc.

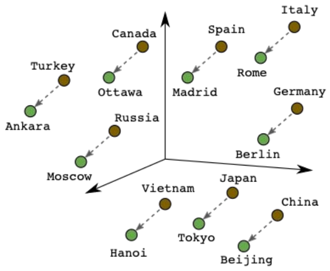
Dynamic Spatio-Temporal Component

Use of Variational AutoEncoders (VAE) .

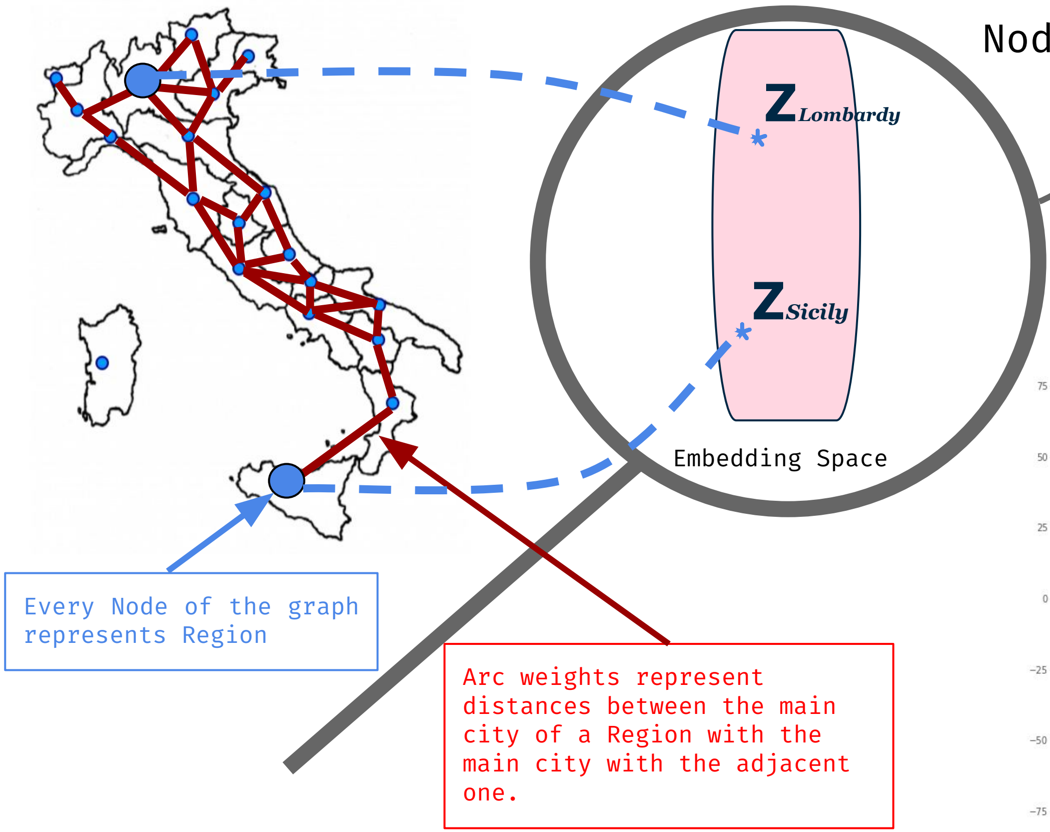
What is the intuition 

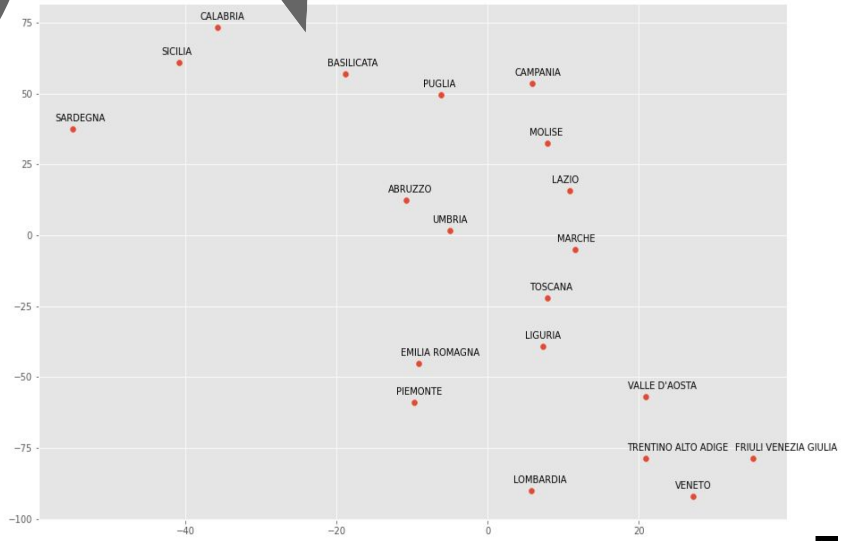
Synthesize an information in a meaningful way





Node2Vec for Spatial Static Component

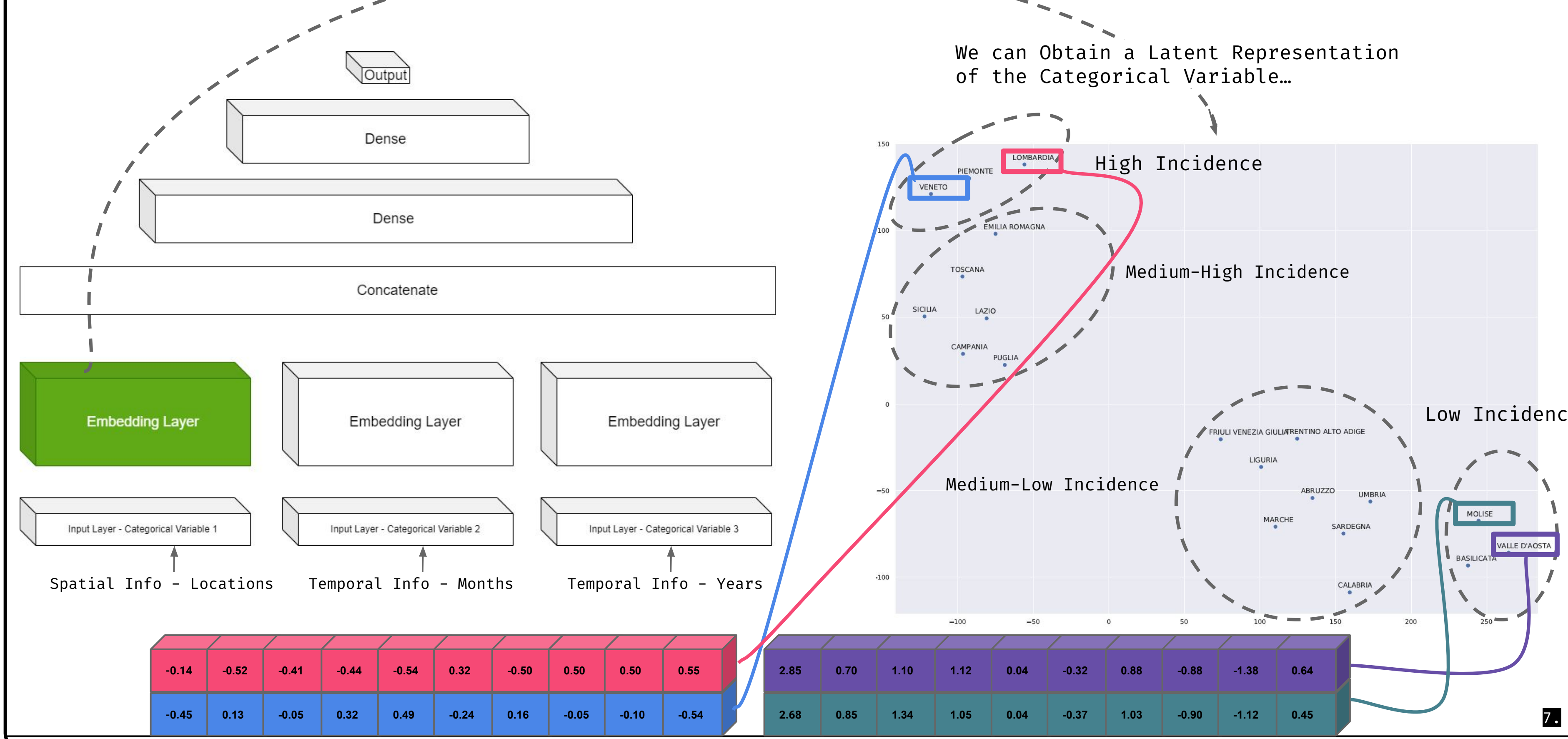




Entity Embedding

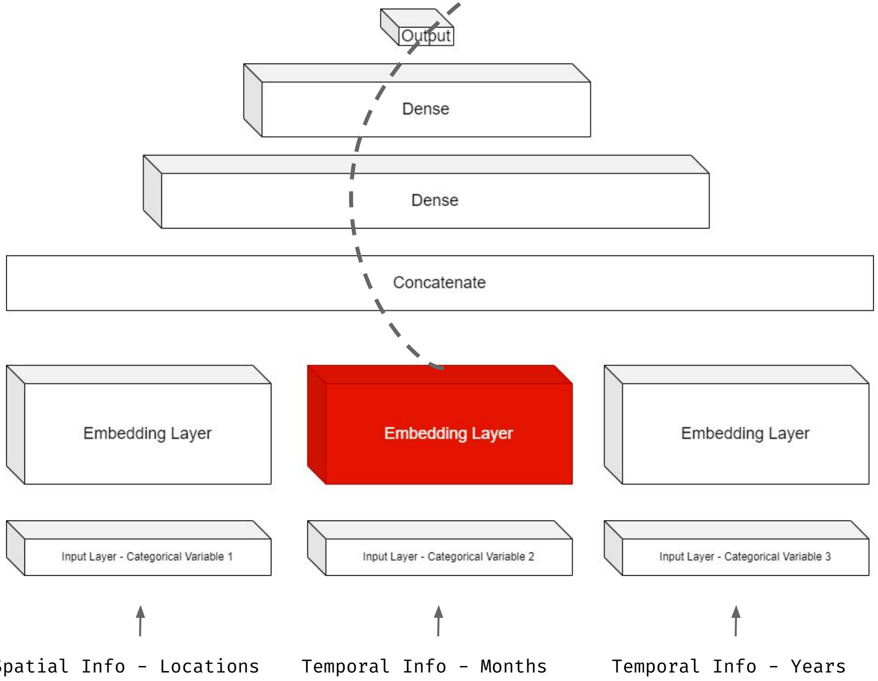
Extracting Weights from Embedding Layers…

We can Obtain a Latent Representation of the Categorical Variable…

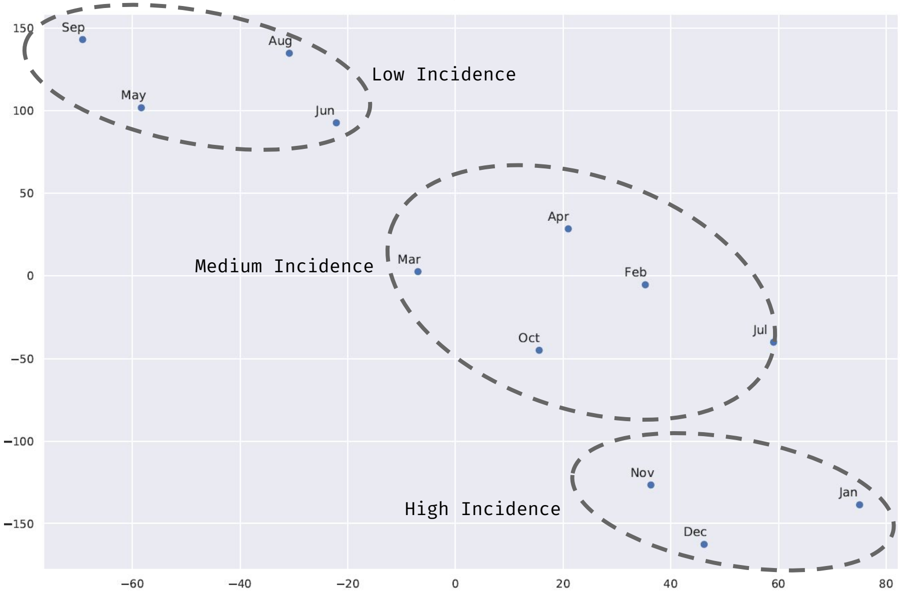


Entity Embedding

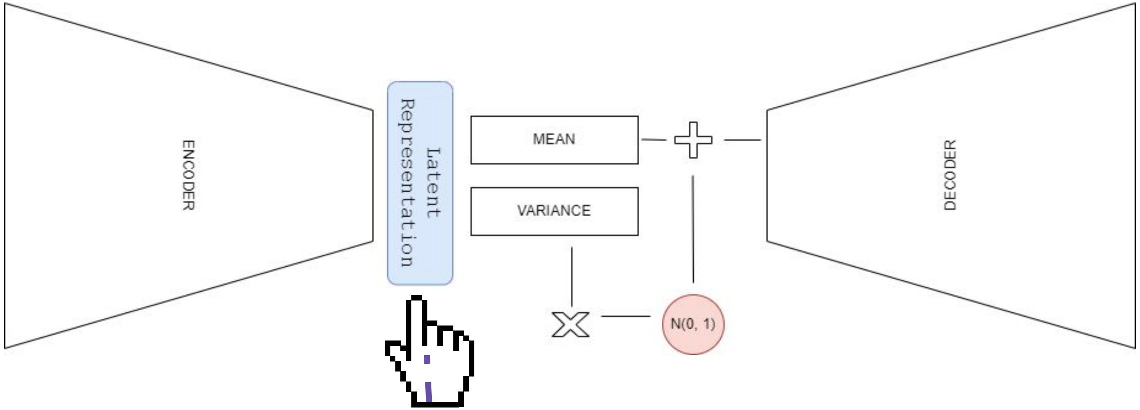
Extracting Weights from Embedding Layers…



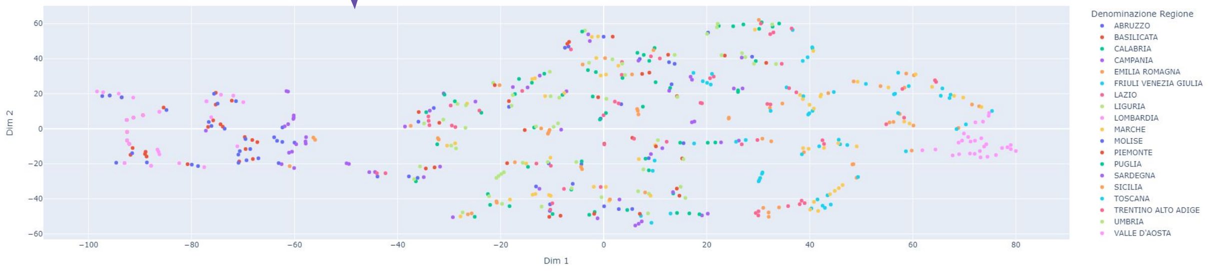
We can Obtain a Latent Representation of the Categorical Variable…



Variational AutoEncoder to Extract Dynamic Spatio-Temporal Embeddings

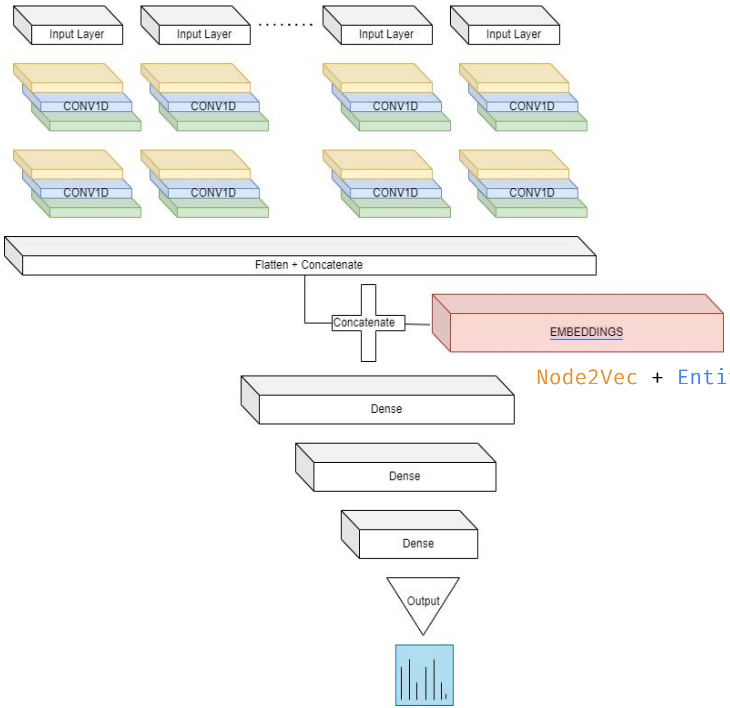


$$loss = ||x - x_{reconstructed}||^2$$



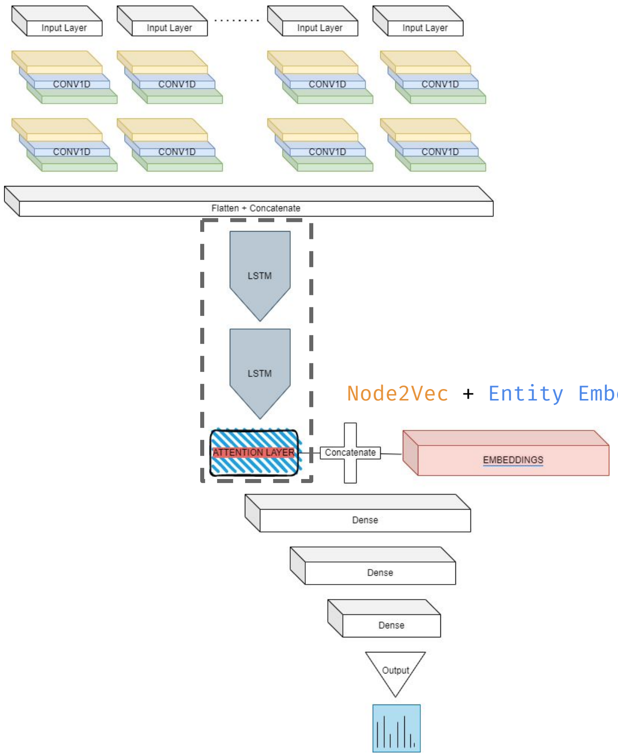
2 Types of BSTNN

BSTNN 1



Node2Vec + Entity Embedding + VAE

BSTNN 2



Node2Vec + Entity Embedding

Main Differences and Similarities Between INLA & BSTNN:

INLA | BSTNN

SPATIAL COMPONENT

INLA:

- Incorporate **spatial dependence** through spatial random effects and/or spatial covariates.

BSTNN:

- Different types of Embeddings (Node2Vec & Entity Embedding) are created

TEMPORAL COMPONENT

INLA:

- AR, MA, ARMA, ARIMA, RW

BSTNN:

- Sequential informations are handled by **LSTM or GRU** network and Attention Layers.
- Additionally, **Temporal Embeddings** are fed as input to the Network.

SMOOTHING

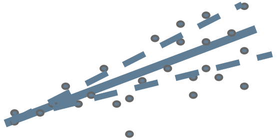
INLA:

- **Splines** can be used to fit a smooth curve to the data. Spline Basis functions are included as **fixed effects** in the model.

BSTNN:

- Convolutional Layers are used as part of a hybrid model that combines convolutional and recurrent layers to smooth time series data.

UNCERTAINTY



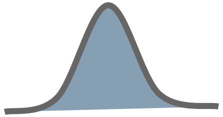
INLA:

- Samples from the posterior predictive distribution can be used to compute 5th and 95th percentiles, which can define the **posterior predictive intervals**.

BSTNN:

- Samples from the posterior predictive distribution can be used to compute 5th and 95th percentiles, which can define the **posterior predictive intervals**.

OUTCOME MODELLING



INLA:

- For **Counting Data** common choices are Poisson, Zero Inflated Poisson, and Negative Binomial.
- For **Continuous Data** Gaussian is the common choice.

BSTNN:

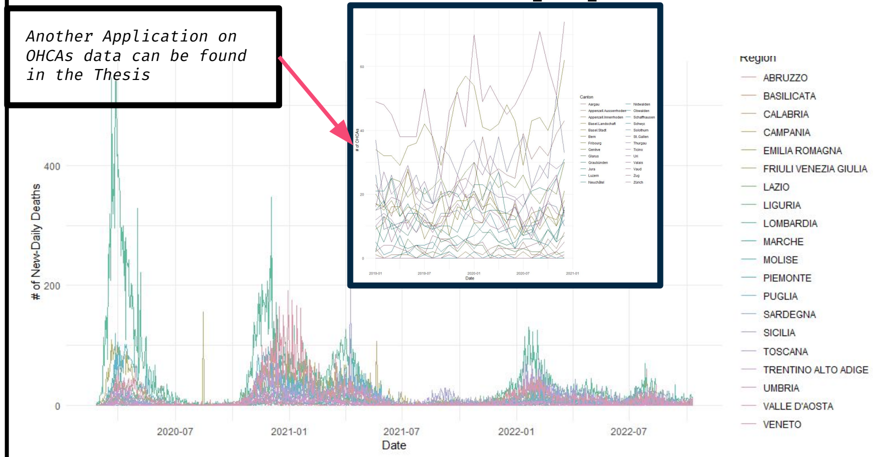
- For **Counting Data** common choices are Poisson, Zero Inflated Poisson, and Negative Binomial, but also mixture of distributions.
- For **Continuous Data** Gaussian is the common choice.

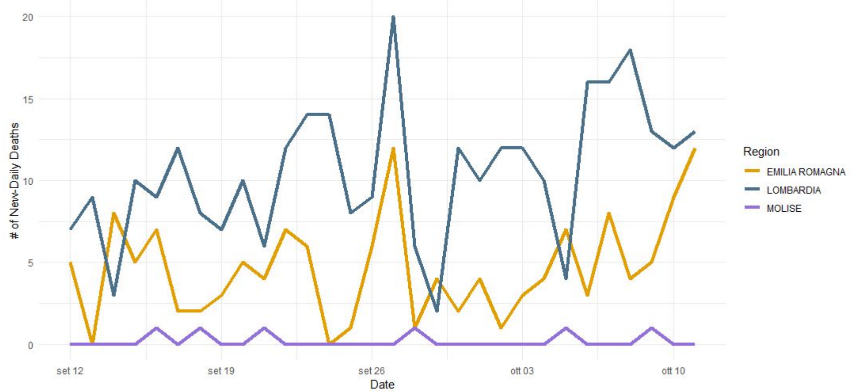
Recap...

	INLA	BSTNN
Spatial Modelling	✓	✓
Temporal & Sequential Modelling	✓	✓
Uncertainty Quantification	✓	✓
Outcome Modelling	✓	✓
Free of assumptions to be met	✗	✓

A Real Application on COVID-19 Data

Another Application on OHCAs data can be found in the Thesis





- Data source comes from the github repository of the **Italian Civil Protection**.
- We focus on New-Daily deceased in each Italian Region. This phenomenon is subject to **high variability** and to strong shocks, for this reason, capturing the signal and purifying it from the noise requires the use of complex models.

2 other Models are used to compare the results w.r.t. the Bayesian Spatio-Temporal approach:

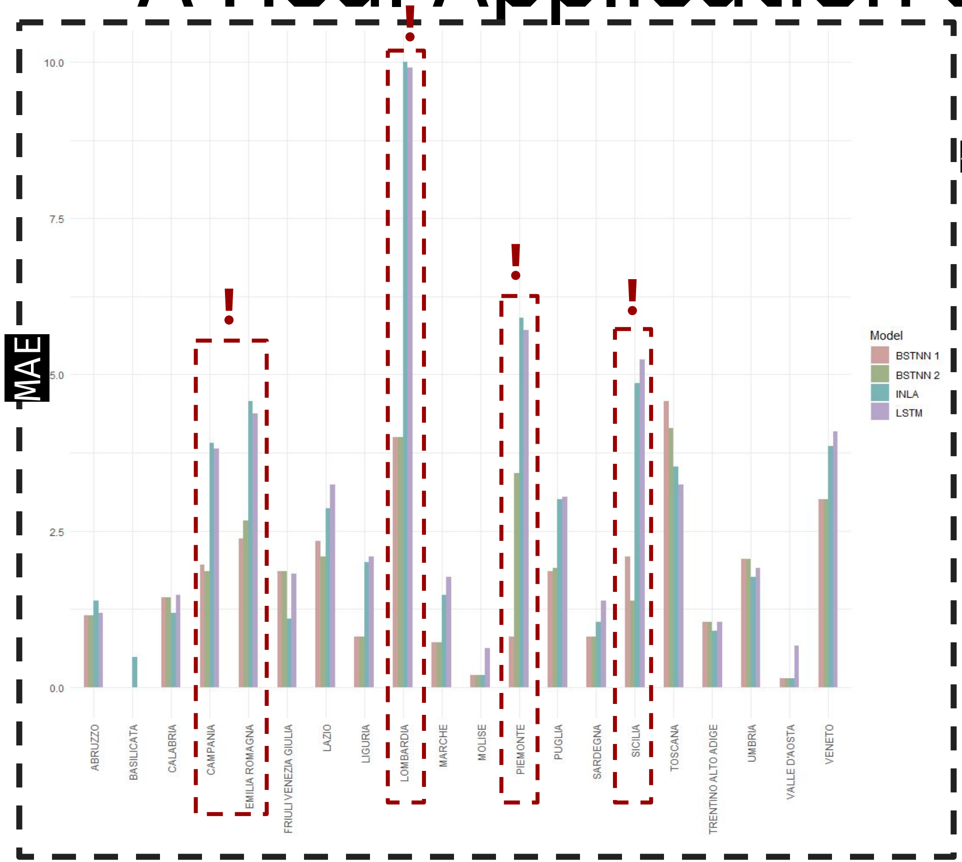
- **INLA**
- **An LSTM implemented for each Region**

These represent the traditional **State-Of-The-Art** models from a Statistical and Machine Learning point of view.

- Performances are evaluated over a variable forecast range (**7, 14 and 21 days**) and it is analyzed how these changes in forecasting ranges impact on the evaluation Metric.
- Mean Absolute Error is selected as evaluation metric for its easy interpretation.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}\left(|y_j - \widehat{y_j}\right)$$

A Real Application on COVID-19 Data



~2x

~2x

~2.5x

~7.5x

~3.5x

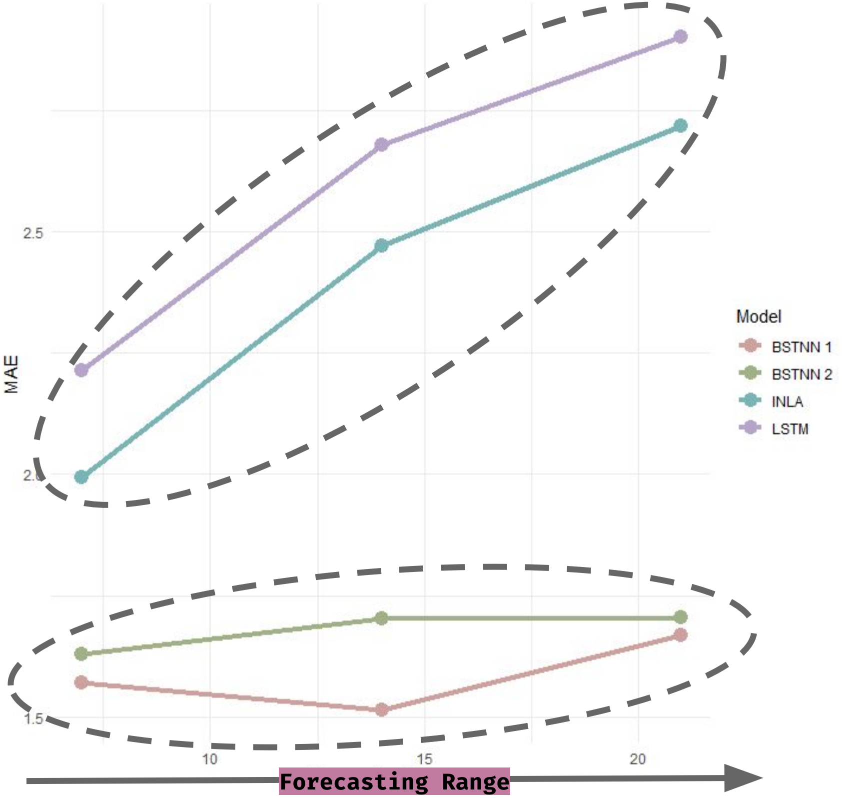
Region	BSTNN 1	BSTNN 2	INLA	LSTM
ABRUZZO	**1.14**	**1.14**	1.38	1.19
BASILICATA	**0.00**	**0.00**	0.48	**0.00**
CALABRIA	1.43	1.43	**1.19**	1.48
CAMPANIA	1.95	**1.86**	3.90	3.81
EMILIA ROMAGNA	**2.38**	2.67	4.57	4.38
FRIULI VENEZIA GIULIA	1.86	1.86	**1.10**	1.81
LATIUM	2.33	**2.10**	2.86	3.24
LIGURIA	**0.81**	**0.81**	2.00	2.10
LOMBARDY	**4.00**	**4.00**	10.00	9.90
MARCHES	**0.71**	**0.71**	1.48	1.76
MOLISE	**0.19**	**0.19**	**0.19**	0.62
PIEDMONT	**0.81**	3.43	5.90	5.71
APULIA	**1.86**	1.90	3.00	3.05
SARDINIA	**0.81**	**0.81**	1.05	1.38
SICILY	2.10	**1.38**	4.86	5.24
TUSCANY	4.57	4.14	3.52	**3.24**
TRENTINO ALTO ADIGE	1.05	1.05	**0.90**	1.05
UMBRIA	2.05	2.05	**1.76**	1.90
AOSTA VALLEY	**0.14**	**0.14**	**0.14**	0.67
VENETO	**3.00**	**3.00**	3.86	4.10

The temporal interval considered for the training set is pretty wide (19’200 observations from 2020-02-25 to 2022-09-20.
Test set is composed by the following 21 days observations (from 2022-09-21 to 2022-10-11).
From a graphical point of view, it is clear that BSTNN tends to predict better where INLA and LSTM have excessively high errors.

A Real Application on COVID-19 Data

As the forecast interval **increases**, the INLA and LSTM forecasts **get worse significantly** (blue and purple line respectively).

As the forecast interval increases, the two BSTNN architectures forecasts get slightly worse (green and red lines) after 21 days, but **much less pronounced** w.r.t.the ones above.



Conclusion & Further Developments

This thesis presented a new deep learning architecture called BSTNN that is well suited for forecasting Spatio-Temporal data.

- The proposed architecture outperforms state-of-the-art Statistics and Machine Learning methods on two different Real World Datasets.

- The proposed architecture has several advantages over existing approaches.
  * Able to account for both the aleatoric and epistemic uncertainties;
  * Model Spatial and Temporal components
  * No need to meet any strict statistical assumption.

What to develop 

- Extend the architecture in order to handle Multivariate Time Series
- A better characterization of the graph from which the Node2Vec embeddings are extracted by introducing new weights on the arcs including:
  - The travel time using different types of vehicles (trains, planes, buses);
  - The number of (travelling) flows from one region to another.

These can be key informations, especially for analyzing an epidemic phenomenon.

Thank you for the Attention

Any Question?

[10] P. Chernyavskiy, M. P. Little, and P. S. Rosenberg, "Spatially varying age–period–cohort analysis with application to us mortality, 2002–2016," Biostatistics, vol. 21, no. 4, pp. 845–859, 2020.

[11] M. Ugarte, T. Goicoa, and A. Militino, "Spatio-temporal modeling of mortality risks using penalized splines," Environmetrics: The Official Journal of the International Environmetrics Society, vol. 21, no. 3-4, pp. 270–289, 2010.

[12] S. Wang, J. Cao, and P. Yu, "Deep learning for spatio-temporal data mining: A survey," IEEE transactions on knowledge and data engineering, 2020.

[13] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," arXiv preprint arXiv:1707.01926, 2017.

[14] N. G. P[illegible] and [illegible] O. Sokolov, "Deep learnin[illegible] for short-term traffic flow predicti[illegible]," Tr[illegible] C: [illegible] Technologies, vol. 79, pp. 1[illegible]17, 20[illegible].

[15] Z. Wu, S. Pa[illegible], G. L[illegible]g, J. Jia[illegible] C. Zhan[illegible], "[illegible] wavenet for deep spatial-temporal graph modeling," arXiv preprint arXiv:1906.001[illegible], 20[illegible].

[16] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," in Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI'19, p. 1907–1913, AAAI Press, 2019.

[17] A. Kapoor, X. Ben, L. Liu, B. Perozzi, M. Barnes, M. Blais, and S. O'Banion, "Examining covid-19 forecasting using spatio-temporal graph neural networks," arXiv preprint arXiv:2007.03113, 2020.

[18] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," in Proceedings of the AAAI conference on artificial intelligence, vol. 34, pp. 1234–1241, 2020.

[19] H. Zhang, S. Li, Y. Chen, J. Dai, and Y. Yi, "A novel encoder-decoder model for multivariate time series forecasting," Computational Intelligence and Neuroscience, vol. 2022, 2022.

[20] S. Arlinghaus, Practical handbook of curve fitting. CRC press, 1994.

[21] D. Lambert, "Zero-inflated poisson regression, with an application to defects in manufacturing," Technometrics, vol. 34, no. 1, pp. 1–14, 1992.

[22] A. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. Driessche, E. Lockhart, L. Cobo, F. Stimberg, et al., "Parallel wavenet: Fast high-fidelity speech synthesis," in International conference on machine learning, pp. 3918–3926, PMLR, 2018.

[23] P. Dellaportas, J. J. Forster, and I. Ntzoufras, "On bayesian model and variable selection using mcmc," Statistics and Computing, vol. 12, no. 1, pp. 27–36, 2002.

[24] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," 1970.

[25] O. Dürr, B. Sick, and E. Murina, Probabilistic Deep Learning: With Python, Keras and TensorFlow Probability. Manning Publications, 2020.

[26] D. T. Chang, "Probabilistic deep learning with probabilistic neural networks and deep probabilistic models," arXiv preprint arXiv:2106.00120, 2021.

[27] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv preprint arXiv:1312.6114, 2013.

[28] C. Blundell, J. Cor[illegible]ise, K. Kavukcuoglu, and D. Wi[illegible]stra, "Weight uncertainty in neural n[illegible]work," in International co[illegible]ence on machine learning, pp. 161[illegible]1622, PMLR, [illegible]015.

[29] Y. Ga[illegible]d Z. Gh[illegible]rani, "[illegible]ut as a bay[illegible]ian [illegible]prox[illegible]: Representing model uncertainty in deep learning," in international conference on machine learning, pp. 1050–1059, PMLR, 2016.

[30] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," The journal of machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.

[31] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in Proceedings of the thirtieth annual ACM symp[illegible] on Theory of [illegible], pp. 604–[illegible]13, 1998.

[32] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[33] X. Rong, "word2vec parameter learning explained," arXiv preprint arXiv:1411.2738, 2014.

[34] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543, 2014.

[35] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," Transactions of the association for computational linguistics, vol. 5, pp. 135–146, 2017.

[36] C. Guo and F. Berkhahn, "Entity embeddings of categorical variables," arXiv preprint arXiv:1604.06737, 2016.

[37] Z. Liu, C. Xiong, M. Sun, and Z. Liu, "Explore entity embedding effectiveness in entity retrieval," in China National Conference on Chinese Computational Linguistics, pp. 105–116, Springer, 2019.

[38] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[39] K. Hornik, "So[illegible]e new results on neural network approximation," Neural networks, vol. 6, no. 8, pp. 1069–1072, 1993.

[40] [illegible] Van der Ma[illegible]en and G. Hinton, "Visualizing data using t-sne.," Journal of machine l[illegible]arning research, vol. 9, no. 11, 2008.

[41] [illegible]. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in Proceedings of the 20th ACM SIGKDD international [illegible]

[illegible] works," in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855–864, 2016.

[44] R. Tarjan, "Depth-first search and linear graph algorithms," SIAM journal on computing, vol. 1, no. 2, pp. 146–160, 1972.

[45] S. Beamer, K. Asanovic, and D. Patterson, "Direction-optimizing breadth-first search," in SC'12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, pp. 1–10, IEEE, 2012.

[46] D. P. Kingma, M. Welling, et al., "An introduction to variational autoencoders," Foundations and Trends® in Machine Learning, vol. 12, no. 4, pp. 307–392, 2019.

[47] Y. LeCun, Y. Bengio, et al., "Convolutional networks for images, speech, and time series," The handbook of brain theory and neural networks, vol. 3361, no. 10, p. 1995, 1995.

[48] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[49] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv preprint arXiv:1412.3555, 2014.

VARIATIONAL INFERENCE

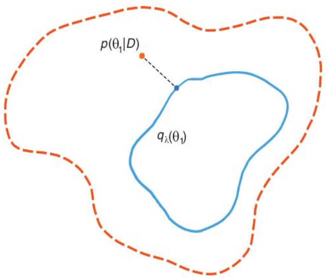
OBJECTS OF INTEREST:

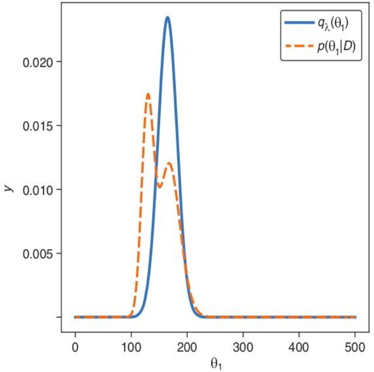
- POSTERIOR DISTRIBUTION $P(w|D)$
- VARIATIONAL POSTERIOR $q(w|\theta)$
- PRIOR $P(w)$
- KULLBACK-LEIBLER DISTANCE

Instead of determining the posterior directly we approximate it with a simple, variational distribution so we want the distance between the variational posterior and the real posterior to be as small as possible.

$$D_{KL}(q(w\,|\,\theta)\,||\,P(w\,|\,D)) =$$
$$= \int q(w\,|\,\theta)\,\log\frac{q(w\,|\,\theta)}{P(w\,|\,D)}\,dw =$$

$$= \int q(w\,|\,\theta)\,log\frac{q(w\,|\,\theta)\,P(D)}{P(D\,|\,w)\,P(w)}\,dw =$$
$$= \log P(D) + D_{KL}(q(w\,|\,\theta)\,||\,P(w)) - E_{q(w\,|\,\theta)}[\log(P(D\,|\,w))]$$



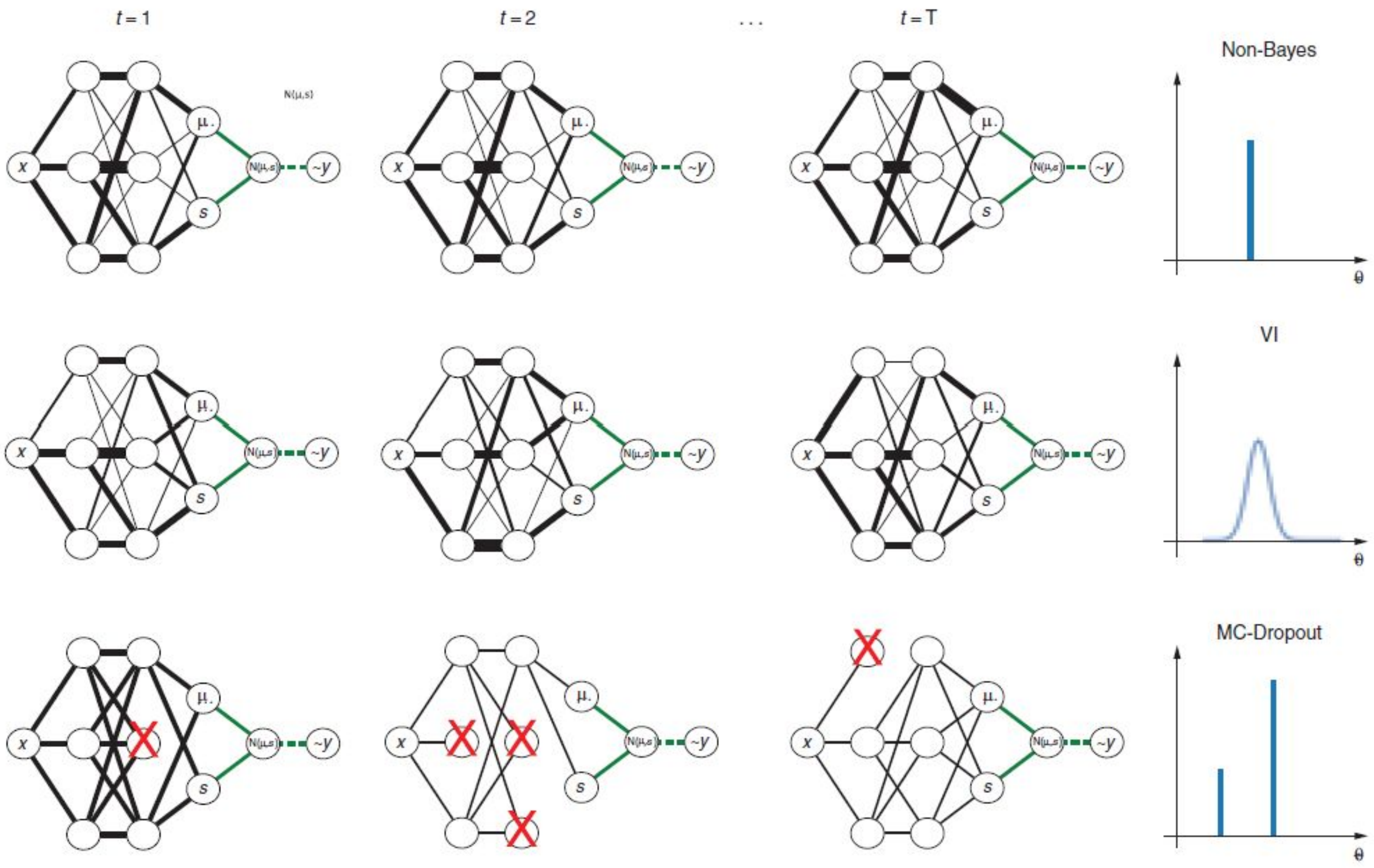


If we consider the data constant,

$$L(\theta\,|\,D) = D_{KL}(q(w\,|\,\theta)\,||\,P(w)) - E_{q(w\,|\,\theta)}[\log(P(D\,|\,w))]$$

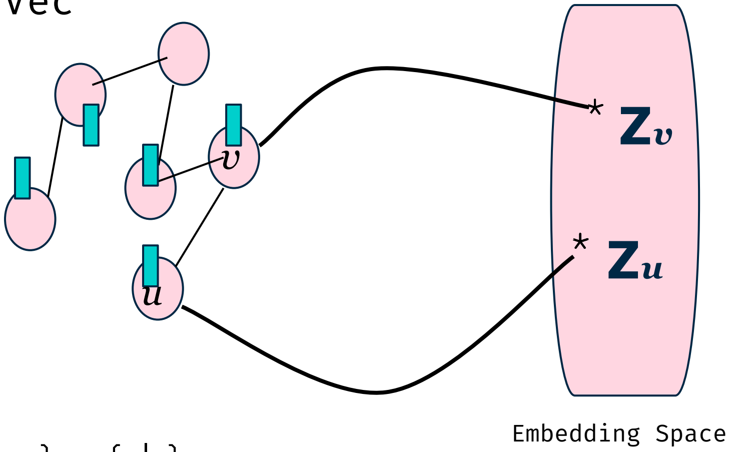
$L(\theta\,|\,D)$ is the loss function we want to minimize

$$\begin{aligned}\log P(D) &= D_{KL}(q(w\,|\,D)\,||\,P(w\,|\,D)) + D_{KL}(q(w\,|\,\theta)\,||\,P(w)) - E_{q(w\,|\,\theta)}[\log(P(D\,|\,w))]\\ &\geqslant E_{q(w\,|\,\theta)}[\log(P(D\,|\,w)) - D_{KL}(q(w\,|\,\theta)\,||\,P(w))] =\\ &\doteq ELBO\end{aligned}$$



t = 1
t = 2
. . .
t = T
Non-Bayes
VI
MC-Dropout
x
μ.
s
N(μ,s)
~y
θ

Node2Vec



$S_G(u, v) = p(u|v)$

$S_E(u, v) = softmax(z)$

Goal: $S_G(u, v) = S_E(u, v)$

How to Explore the network?

1. Depth-first search (DFS) - Explore Global Representation
2. Breadth-first search (BFS) - Explore Local Representation

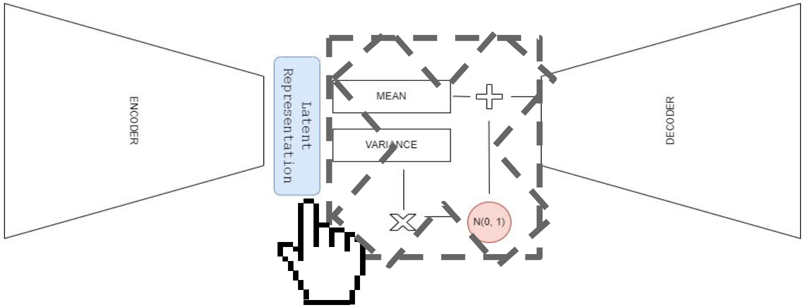
Two parameters p and q for probability of returning to the starting node (local exploration) and probability of moving away from the starting node (global exploration)

Similarity between nodes u and v is defined as the probability of visiting u if do a random walk on graph starting at node v

Variational AutoEncoder

What's the difference between a VAE and an AE?

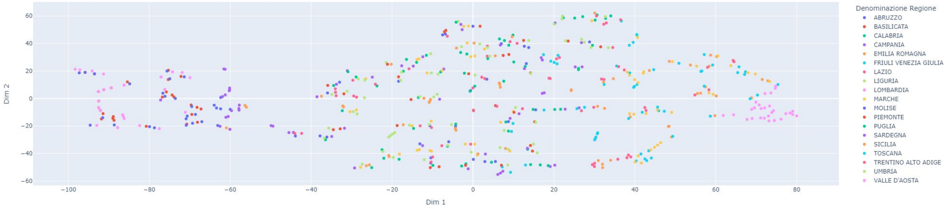
Let's say we have an AutoEncoder with two latent variables and we draw samples randomly and get two samples of 0.4 and 1.2. We then send them to the decoder for data reconstruction. In a VAE, these samples don't go to the decoder directly. Instead, they are used as a mean and a variance of a Gaussian Distribution, and the network use them to draw samples from the gaussian distribution to be sent to the decoder for data reconstruction purpose.



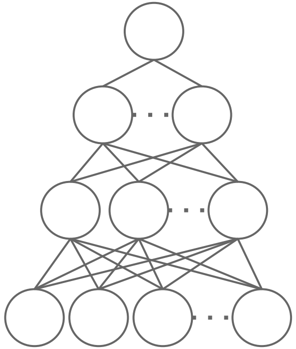
Some of the main advantages of regularization in VAEs include:

- Improved generalization: By encouraging the VAE to learn more general features, regularization can help the VAE to generalize better to new data points that it has not seen before. This can lead to better performance on the test set, and make the VAE more robust to changes in the data distribution.
- Reduced overfitting: Overfitting occurs when a model becomes too closely tied to the training data, and is unable to generalize to new data points. By using regularization, VAEs can avoid overfitting and achieve better performance on the test set.
- Better interpretability: VAEs that are regularized can be more interpretable because the latent space is more likely to have a clear meaning.
- Improved stability: Regularization can also help to improve the stability of VAEs, by encouraging the model to learn more robust and stable features. This can make the VAE less sensitive to small variations in the training data, and can help to prevent the model from learning features that are not meaningful or relevant to the task.

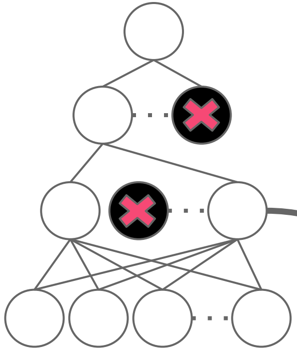
$$loss = ||x - x_{reconstructed}||^2 + KL\left[N\left(\mu_x, \sigma_x\right), N(0, 1) \right]$$



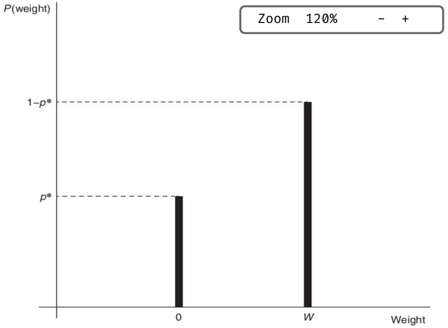
MC DROPOUT



Standard Neural Network



After applying Dropout



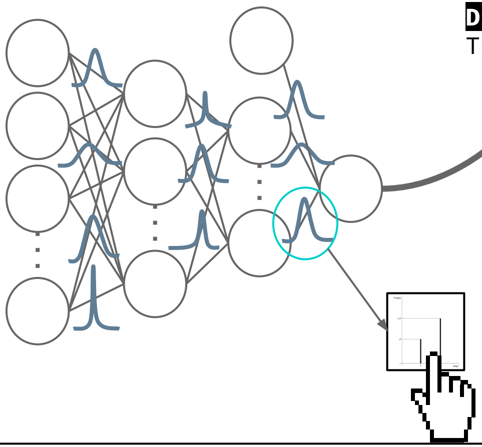
When using MC Dropout, we end up with a weight distribution, but the weight distribution consists of only two values: **0 or W**.

INTUITION:

- TREAT MANY DIFFERENT NETWORKS AS MC SAMPLES FROM THE SPACE OF ALL AVAILABLE MODELS.

TO DO THAT

If we **Turn On Dropout** during Testing Step

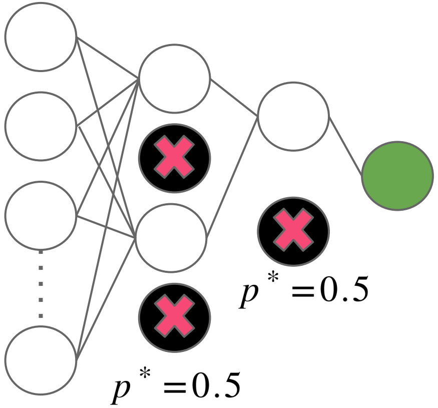


We can then combine the dropout predictions to a **Bayesian predictive distribution**:

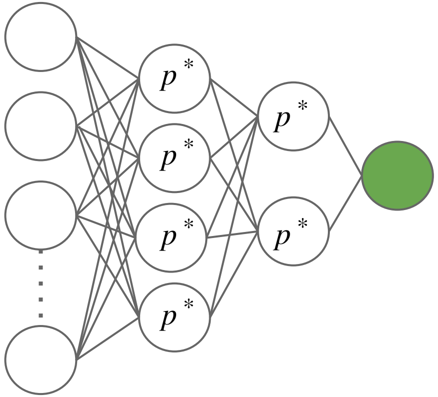
$$p(y \mid x_{test}, D) = \frac{1}{T} \sum_{t}^{T} p(y \mid x_{test}, w_t)$$

Classic Dropout … vs. MC Dropout

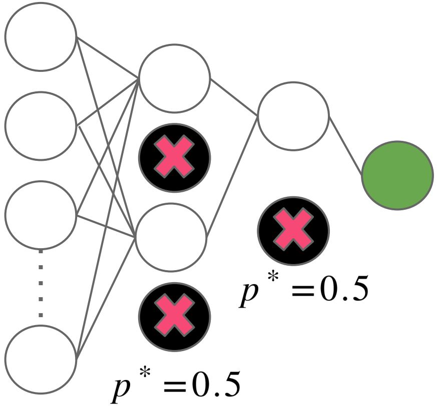
Dropout during Training



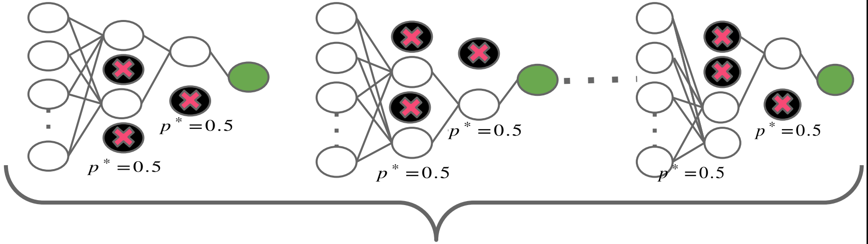
Dropout during Predictions



Dropout during Training



Dropout during Predictions



N Different Predictions for each Test Observation