IA353 – Redes Neurais (1s2021)
Exercícios de Fixação de Conceitos 2 – EF 2
Atividade Individual – Peso 6
Data de entrega dos resultados solicitados: 27/06/2021

Questão 5)

Tomando a base de dados MNIST, treinar um *autoencoder* e constatar o sucesso da tarefa de codificação e de decodificação (mesmo que ainda com algumas imperfeições), indicando que duas variáveis latentes podem ser suficientes para codificar a essência de toda a variedade exibida pelas versões manuscritas de 10 dígitos. Por fim, explorar o *manifold* gerado no gargalo do *autoencoder*. Para tanto, como um primeiro passo, execute o notebook fornecido pelo professor (**Q5.ipynb**) e procure compreender o que está sendo feito em cada trecho de código. Atividades práticas:

1. Com base no desempenho obtido com a proposta fornecida, procure melhorar a distribuição das classes ao longo do *manifold*, atuando na arquitetura do *autoencoder* e nos parâmetros de treinamento. Forneça o novo notebook gerado, com a denominação **([seu_RA]Q5_1.ipynb**).
2. Apresente um notebook que aborde algum outro problema a ser resolvido com *autoencoder* (pode ser a versão variacional), executando e comentando as principais etapas envolvidas na solução. Devem ser apresentados resultados gráficos que comprovem o bom desempenho. Forneça o novo notebook gerado, com a denominação **([seu_RA]Q5_2.ipynb**).

Questão 6)

São 3 partes envolvidas nesta questão. Todas elas vinculadas à predição de séries temporais. Para mais detalhes acerca do assunto e das etapas de pré-processamento envolvidas, favor consultar o apêndice deste roteiro de estudos.

Parte 1 (Peso 0,4): Execute o notebook **Q6P1_NYSP.ipynb**, constatando as funcionalidades envolvidas em cada trecho de código e analisando os resultados envolvidos. Atividades práticas:

6.1.1. Forneça o notebook devidamente executado, contendo, ao final, um resumo geral do que foi trabalhado em cada seção.

6.1.2. Com o foco no desempenho do preditor linear, procure apresentar argumentos capazes de sustentar o fato de a predição proposta corresponder basicamente a uma versão atrasada da própria série temporal.

6.1.3. Embora seja possível obter, com os modelos de predição não-lineares, um desempenho similar àquele do preditor linear (isso não é solicitado ao aluno, no entanto), explique por que o preditor linear é tão competente nesta tarefa de predição específica.

6.1.4. Procure justificar também por que alguns preditores não-lineares estão fornecendo uma predição que, aparentemente, falha apenas no *offset* da predição, ou seja, acompanham o comportamento da predição, mas erram no seu valor médio.

Parte 2 (Peso 0,4): Execute o notebook **Q6P2_NLTS.ipynb**, que busca realizar predições de uma série temporal sintética, gerada pelo professor e sem a disponibilidade do processo gerador da série temporal, que garantidamente apresenta comportamento não-linear e estacionário. Atividades práticas:

6.2.1. Forneça o notebook devidamente executado, contendo, ao final, um resumo geral do que foi trabalhado em cada seção. Forneça, por exemplo, o número de atrasos da série temporal que está sendo considerado como entrada dos preditores e como está sendo feita a regularização dos modelos, quando for o caso.

6.2.2. Dadas as características da série temporal, explique o motivo pelo qual o preditor linear saiu de primeiro colocado em desempenho, na Parte 1, para último colocado nesta Parte 2.

6.2.3. Escolha um dos preditores e implemente a predição de múltiplos passos à frente (*several time steps ahead*), explicando como funciona a sua metodologia.

6.2.4. Sem a necessidade de implementações práticas, explique que modificações metodológicas você faria caso se constatasse que a série temporal é não-estacionária, mas com uma variação lenta de suas propriedades estatísticas ao longo do tempo.

Parte 3 (Peso 0,2): Tome o preditor linear e ao menos um dos preditores não-lineares implementados e resolva um problema de predição de séries temporais da literatura (predição um passo à frente), à sua escolha, mas envolvendo dados reais e não dados sintéticos. Apresente o notebook com os resultados obtidos, justificando suas decisões metodológicas e analisando os resultados. Se possível, inclua comparações com a literatura. Forneça tudo num notebook **Q6P3.ipynb**.

Questão 7)

Esta questão aborda conceitos de interpretabilidade em aprendizado de máquina, evidenciando dois paradigmas de grande relevância e focando num subconjunto de técnicas, dentre várias possibilidades que vêm sendo propostas na literatura. A busca por modelos de aprendizado interpretáveis sempre esteve na alça de mira das pesquisas em aprendizado de máquina, mas até recentemente havia uma predominância de foco em acurácia e se defendia a existência de um compromisso entre acurácia e interpretabilidade, de modo que a busca por mais acurácia tenderia a gerar modelos menos interpretáveis, da mesma forma que modelos mais interpretáveis não conseguiriam atingir os mesmos níveis de acurácia que modelos "caixa-preta". As últimas conquistas voltadas para interpretabilidade em aprendizado de máquina, no entanto, demonstram que acurácia e interpretabilidade podem caminhar juntas, trazendo maior confiabilidade para aplicações de inteligência artificial.

Tomando o mesmo problema de classificação de dados da base MNIST, o objetivo desta questão é analisar a interpretabilidade de uma RNA treinada. Para tanto, vamos utilizar o código a seguir para treinar uma RNA.

```
import keras

mnist = keras.datasets.mnist
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_train, x_test = x_train / 255.0, x_test / 255.0

model = keras.models.Sequential()
model.add(keras.layers.Conv2D(32, kernel_size=(3, 3),
 activation='relu',input_shape=(28, 28, 1)))
model.add(keras.layers.Conv2D(64, (3, 3), activation='relu'))
model.add(keras.layers.MaxPooling2D(pool_size=(2, 2)))
model.add(keras.layers.Dropout(0.25))
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(128, activation='relu'))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Dense(10, activation='softmax'))

model.get_config()

model.compile(optimizer='adam',
 loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5)
evaluation = model.evaluate(x_test, y_test)

model.save('mnist_model.h5')
```

7a)

Utilize a biblioteca *innvestigate* [1] para analisar as predições da RNA e comparar diferentes métodos de explicação [3]. Uma explicação é a coleção de características do domínio interpretável, que contribuíram para que uma dada amostra produzisse uma decisão (de classificação ou regressão, por exemplo) [3]. Utilize os seguintes métodos da biblioteca *innvestigate*: *Gradient*, *SmoothGrad*, *DeepTaylor*, *LRPAlphaBeta*, *LRPEpsilon*, *LRPZ*. Um código que pode servir de ponto de partida é fornecido a seguir. Escolha 6 imagens aleatórias de 3 classes distintas para analisar, portanto teremos 2 imagens por classe. Escolha parâmetros adequados para os métodos *LRPAlphaBeta* e *LRPEpsilon* (use os mesmos parâmetros para as 6 imagens escolhidas). A Figura 1 apresenta um modelo de como os resultados devem ser exibidos. Analise o resultado obtido.

```
import keras
import innvestigate
import matplotlib.pyplot as plot

mnist = keras.datasets.mnist
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train = x_train.reshape(x_train.shape[0], 28, 28, 1)
x_test = x_test.reshape(x_test.shape[0], 28, 28, 1)
x_train, x_test = x_train / 255.0, x_test / 255.0

model = keras.models.load_model('mnist_model.h5')
model_wo_sm = innvestigate.utils.keras.graph.model_wo_softmax(model)

imagem = x_test[0:1]
plot.imshow(imagem.squeeze(), cmap='gray', interpolation='nearest')

analyzer = innvestigate.analyzer.LRPEpsilon(model=model_wo_sm,
 epsilon=1)
analysis = analyzer.analyze(imagem)
plot.imshow(analysis.squeeze(), cmap='seismic',
 interpolation='nearest')
```

7b)

Utilize a biblioteca keras-vis [2] para interpretar as predições da RNA para cada uma das 10 classes. Neste caso, procuramos por um padrão de entrada que produza uma resposta máxima do modelo para uma métrica de interesse [3]. Um código que pode servir de ponto de partida é fornecido a seguir. Escolha valores adequados para os parâmetros que determinam os pesos da *Total variation* e *L-p norm*. Veja dicas em:

https://github.com/raghakot/keras-vis/blob/master/examples/mnist/activation_maximization.ipynb

Plote os gráficos das imagens obtidas para cada uma das 10 classes e analise o resultado.

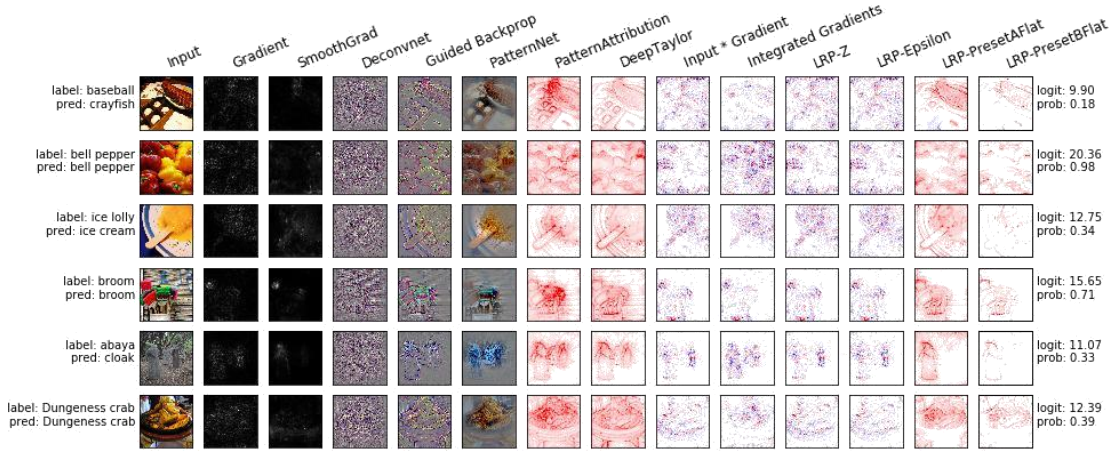


Figura 1 – Exemplo de como exibir os resultados da Questão 7a. Fonte: https://github.com/ albermax/innvestigate

```
import keras
from vis.visualization import visualize_activation
from vis.utils import utils
import matplotlib.pyplot as plot

model = keras.models.load_model('mnist_model.h5')
layer_idx = utils.find_layer_idx(model, 'dense_2')
model.layers[layer_idx].activation = keras.activations.linear
model = utils.apply_modifications(model)

filter_idx = 9
img = visualize_activation(model, layer_idx,
 filter_indices=filter_idx, input_range=(0., 1.), verbose=True,
 max_iter=1000, tv_weight=1., lp_norm_weight=0.)
plot.imshow(img.squeeze(), cmap='seismic', interpolation='nearest')
```

Referências bibliográficas

[1] M. Alber, S. Lapuschkin, P. Seegerer, M. Hägele, K. T. Schütt, G. Montavon, W. Samek, K.-R. Müller, S. Dähne, P.-J. Kindermans, **innvestigate neural networks!**, preprint arXiv:1808.04260, 2018.

[2] R. Kotikalapudi and contributors, **keras-vis**. https://github.com/raghakot/keras-vis, 2017.

[3] G. Montavon, W. Samek, K.-R. Müller, **Methods for interpreting and understanding deep neural networks**, Digital Signal Processing, vol. 73, pp. 1-15, 2018.

Questão 8)

Aprendizado por reforço do tipo *Deep Q-Learning* para definir percursos em labirintos.
Estude o notebook **Q8.ipynb**, procurando compreender o que está sendo feito em cada trecho de código. Para tanto, é muito relevante acompanhar atentamente as explicações em https://www.samyzaf.com/ML/rl/qmaze.html. Atividades práticas:

3. Execute o notebook para os dois labirintos propostos (execuções independentes, usando um de cada vez), apresentando os resultados do treinamento até a convergência, além de 2 percursos para cada caso de estudo (com condições iniciais distintas), obtidos com a rede neural treinada.
4. Para esses dois casos de estudo, apresente os 4 valores da função Q-valor para 3 estados representativos do ambiente, produzidos pela rede neural treinada.
5. Explique como é definida a função de erro quadrático médio usada no treinamento.
6. Explique como é trabalhada a técnica de *experience replay*.

Questão 9)

O treinamento de redes neurais adversárias foi proposto em 2014:
I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, **Generative adversarial nets**, Advances in Neural Information Processing Systems, pp. 2672-2680, 2014.
no contexto de síntese de máquinas generativas, e foi considerada a ideia mais interessante em aprendizado de máquina da última década, tendo já recebido inúmeras

contribuições e extensões na literatura. As GANs conseguem conduzir o processo de treinamento a desempenhos nunca antes vislumbrados em aprendizado de máquina, permitindo afirmar que as redes neurais generativas podem capturar "qualquer" distribuição exibida pelos dados de treinamento, habilidade esta que torna "ilimitado" o potencial de aplicação dessas máquinas de aprendizado. Evidentemente, a exploração de todo este potencial requer muita memória e muito processamento, algo que não será exigido neste curso. Inicialmente, esta atividade envolve a execução completa do treinamento de uma GAN para reproduzir dígitos manuscritos, a partir da base MNIST. Para isso, vamos utilizar a biblioteca Keras-GAN disponível em:

https://github.com/eriklindernoren/Keras-GAN

O objetivo é conseguir executar o exemplo disponível em:

https://github.com/eriklindernoren/Keras-GAN/blob/master/gan/gan.py

que já foi organizado no notebook **Q9.ipynb**. O código executa o treinamento por 30.000 épocas e salva exemplos de imagens a cada 1000 épocas. Execute o código e exiba: a primeira imagem obtida, e imagens obtidas após 1000, 10000, 20000 e 30000 épocas. Por fim, adapte o código ou use algum outro código capaz de treinar uma máquina generativa para alguma outra base de imagens da literatura, apresentando os resultados de forma similar ao caso da base MNIST.

Questão 10)
Processamento de Linguagem Natural – *Word Embedding*, usando a implementação GENSIM do Word2Vec e uma base de reviews de hotéis pelo mundo, chamada OpinRank (https://kavita-ganesan.com/entity-ranking-data/#.XwbHbOWSlPY).
O primeiro passo é instalar o GENSIM:

conda install -c conda-forge gensim

A seguir, copie o arquivo GZIP fornecido pelo professor em algum diretório de trabalho. O *default* no notebook é C:/IA353.
Explique como funcionam Word2Vec e t-SNE (*t-Distributed Stochastic Neighbor Embedding*).
Execute o notebook **Q10.ipynb** e comente / interprete os resultados obtidos.

Observações gerais:

- Os entregáveis devem ser enviados para o e-mail [vonzuben@dca.fee.unicamp.br] com Assunto [IA353 – EF2] até 23h59 do prazo final. Aguarde um retorno por e-mail com a confirmação de recebimento. Entregas atrasadas terão um desconto linear de 1/7 da nota por dia de atraso, de modo que após 7 dias a nota é zerada.

Apêndice: Conceitos básicos de series temporais

1 Introdução

Em qualquer ramo de atuação profissional, percebe-se um crescimento acentuado na demanda por predições e detecção de tendências junto a variáveis de interesse. Ferramentas computacionais capazes de fornecer predições automáticas acerca de valores futuros de certas variáveis que estão sendo monitoradas já fazem parte do dia-a-dia de muitas empresas privadas e órgãos governamentais, e têm contribuído para o

sucesso na definição de políticas estratégicas, em processos de tomada de decisão e em todo tipo de planejamento de curto e médio prazo.

Predição de séries temporais é uma área de pesquisa muito ampla, com desdobramentos na Estatística, na Computação e na Matemática. Serão tratados neste EFC3 apenas alguns aspectos básicos e fundamentais, visando fazer com que o aluno se familiarize com técnicas de síntese de preditores lineares e não-lineares, a partir apenas de dados amostrados das respectivas séries temporais. Outras variáveis de apoio e múltiplas séries correlacionadas poderiam ser consideradas, mas vamos nos restringir ao caso de uma única série temporal por vez. Iremos nos restringir também ao caso de modelos autorregressivos e não iremos recorrer a nenhum outro tipo de tratamento da série temporal, além de aplicar uma operação de escalamento para um intervalo de interesse. Um aprofundamento do estudo pode ser buscado junto à literatura recomendada.

2 Séries temporais e a tarefa de predição

Uma série temporal é dada pelos valores ao longo do tempo de uma variável de interesse, como em:

- ✓ Atividades vitais ou funções orgânicas de um indivíduo;
- ✓ Índices econômicos;
- ✓ Índices sociais;
- ✓ Variáveis climáticas;
- ✓ Variáveis de ecossistemas;
- ✓ Monitoramento de operação de plantas industriais;
- ✓ Monitoramento de processos químicos e físicos.

A frequência de amostragem depende da aplicação e geralmente é fixa. Os intervalos de amostragem mais comuns são ano, mês, semana, dia e hora.

Sendo a predição uma estimativa de valores futuros a partir do conhecimento do histórico de uma variável ou múltiplas variáveis até o presente, surgem algumas questões:

- ✓ O histórico de uma variável até o presente é capaz de auxiliar na predição do seu comportamento futuro?
- ✓ Como geralmente a variável de interesse tem seu comportamento atrelado a uma grande quantidade de fatores e a uma complexa rede de inter-relações, como é possível predizer seu comportamento sem modelar os fenômenos complexos que regem o comportamento da variável e sem monitorar outras variáveis que influenciam nesse comportamento? Exemplo: Como predizer a vazão de um rio com base apenas em seu histórico, sem levar em conta outras variáveis como a vazão dos afluentes, o nível de chuvas na cabeceira do rio, o consumo de água do rio para irrigação agrícola e o grau de assoreamento do leito do rio, dentre outros fatores?

Existem séries temporais que não admitem predição, seja pela independência estatística intrínseca entre o que já se conhece da série e o valor futuro que se pretende predizer, seja pelo fato de que aspectos relevantes do processo estão sendo

negligenciados, como em séries multidimensionais em que alguma(s) variável(is) não está(ão) sendo monitorada(s). Nesses casos, geralmente as iniciativas de predição tendem a produzir resultados decepcionantes.

Por outro lado, existem séries temporais que apresentam uma dependência estatística significativa entre o histórico da série e o valor futuro a ser predito. Nesses casos, as predições tendem a produzir resultados de predição de alta qualidade, mesmo que não se conheçam especificidades do processo que gera a série temporal.

Esta dependência estatística (entre o histórico da série e o valor a ser predito) pode ser estimada a priori. Com isso, antes mesmo de se iniciar o projeto do preditor já é possível ter uma boa noção de como pode vir a ser o seu desempenho naquela tarefa de predição. Além disso, análises de dependência estatística auxiliam na definição da estrutura do preditor, mais especificamente na definição do vetor de regressão, ou seja, o vetor de entrada do preditor, também chamado de linha de derivação de atrasos. Para tanto, geralmente são respondidas duas perguntas:

- Qual é o tamanho da janela de valores passados a serem considerados na entrada do preditor?
- Quais valores passados dentro desta janela devem ser considerados? Todos eles ou só alguns deles?

Coeficientes de correlação linear (contextualizados por Francis Galton em 1885 e devidamente formalizados por Karl Pearson em 1895) (RODGERS & NICEWANDER, 1988; STIGLER, 1989) são muito utilizados para responder a essas perguntas, mas cabe salientar que eles supõem o uso subsequente de preditores lineares. Um índice mais consistente, capaz de capturar dependências lineares e não-lineares, é a informação mútua, derivado da Teoria de Informação (COVER, 1991; REZA, 1994; WESTIAN, 1990).

3 Metodologia

As primeiras tentativas no campo da predição de séries temporais foram efetuadas nos anos 20, quando YULE (1927) aplicou um modelo autorregressivo linear no estudo de manchas solares. Nos anos 50, DOOB (1953) prosseguiu a investigação com a análise teórica de séries temporais estacionárias. Já nos anos 70, foram propostas as técnicas e metodologias que obtiveram maior destaque a partir de então, reunidas no trabalho de BOX & JENKINS (1976).

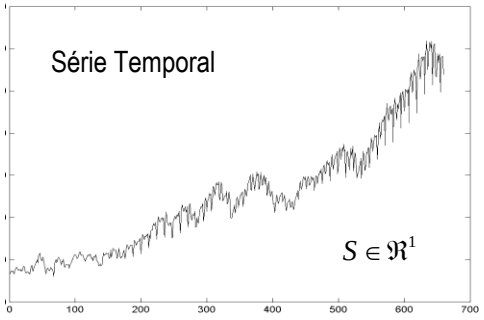
Os métodos de Box & Jenkins baseiam-se na proposição de que o valor atual da série temporal é a combinação de p valores precedentes e q impactos aleatórios anteriores, mais o impacto atual. Os p valores precedentes formam o *componente autorregressivo* e os q impactos prévios formam o *componente de média móvel* da série. Obtêm-se assim os bem conhecidos modelos ARMA (do inglês *autoregressive moving average*). A modelagem de uma série temporal tem por objetivo, então, a determinação dos valores de p e q, seguida da estimação dos respectivos coeficientes da combinação linear. Gera-se, assim, um problema de regressão linear que tem como uma de suas formulações aquela que leva à solução de quadrados mínimos.

Nos últimos anos, considerável atenção tem sido dedicada a métodos alternativos para o estudo de séries com dependências não-lineares, destacando-se a utilização de redes

neurais artificiais. O emprego das arquiteturas MLP e RBF, ao longo das últimas décadas, trouxe resultados muito positivos no campo da predição de valores futuros em séries temporais, em virtude do caráter essencialmente não-linear dessas estruturas.

Para o emprego de uma rede neural artificial como preditor de um passo à frente, é necessário definir quais valores passados da série serão utilizados na definição da entrada da rede neural. Feito isso, o problema de síntese do preditor se transforma em um problema de treinamento supervisionado, onde o que se deseja é obter um mapeamento multidimensional não-linear de entrada-saída, como indicado na sequência de passos abaixo.

Passo 1: Obter a série temporal, ou seja, os valores históricos da variável a ser predita um passo à frente. Se necessário, faça um escalamento dos dados, evitando que o intervalo de excursão dos valores seja qualquer. Outros tipos de pré-processamento, como diferenciar a série temporal, também podem ser considerados visando eliminar tendências e sazonalidades, por exemplo.



$$S = s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8 \ldots, s_N$$

Passo 2: Definir quais valores passados da série serão considerados na predição. Suponha aqui que L valores passados consecutivos sejam considerados. Com isso, monte a tabela a seguir, a qual retrata o comportamento desejado do preditor.

$\boldsymbol{s}_{t-L+1}$	$\boldsymbol{s}_{t-L+2}$	$\cdots$	$\boldsymbol{s}_t$	$\boldsymbol{s}_{t+1}$
s_1	s_2	$\cdots$	s_L	s_{L+1}
s_2	s_3	$\cdots$	s_{L+1}	s_{L+2}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_{N-L}	s_{N-L-1}	$\cdots$	s_{N-1}	s_N
$\Downarrow$	$\Downarrow$		$\Downarrow$	$\Downarrow$
X_1	X_2		X_L	Y

Passo 3: Separe os dados da tabela acima em 3 conjuntos: conjunto de treinamento, conjunto de validação e conjunto de teste. Não é necessário que a separação seja sequencial, mas a separação sequencial é mandatória no caso de redes neurais recorrentes, razão pela qual será adotada aqui.

Passo 4: Treine a rede neural com o conjunto de treinamento (ela vai produzir um mapeamento do $\Re^L$ para o $\Re^1$) e pare o treinamento quando for atingido o valor mínimo do erro quadrático médio para o conjunto de validação.

Passo 5: Avalie o preditor recém-obtido junto aos dados de teste.

4 Agradecimento

Algumas figuras e trechos de texto utilizados neste apêndice são de autoria de Wilfredo Jaime Puma Villanueva, que realizou a sua pós-graduação junto ao Programa de Pós-Graduação da FEEC/Unicamp.

5 Referências

BOX, G. E. P. & JENKINS, G. M. (1976). *Time Series Analysis, Forecasting and Control*. Holden Day.

BRADLEY, E. & KANTZ, H. (2015). *Nonlinear time-series analysis revisited*. arXiv:1503.07493v1.

COVER, T. M. & THOMAS, J. A. (1991) *Elements of Information Theory*, Wiley.

DOOB, J. (1953). *Stochastic Processes*. Wiley.

PRECHELT L. (1997). Early Stopping - but when?, Technical Report URL: http://wwwipd.ira.uka.de/~prechelt/Biblio/stop_tricks1997.ps.gz

PRECHELT L. (1998). Automatic early stopping using cross validation: Quantifying the criteria. *Neural Networks*, vol. 11, no 4, pp. 761-767.

REZA, F. M. (1994) *An Introduction to Information Theory*, Dover.

RODGERS, J. L. & NICEWANDER, W. A. (1988) Thirteen ways to look at the correlation coefficient. *The American Statistician*, vol. 42, no. 1, pp. 59-66.

STIGLER, S. M. (1989). Francis Galton's Account of the Invention of Correlation. *Statistical Science*, vol. 4, no. 2, pp 73-79.

WEIGEND, A. S. & GERSHENFELD, N. A. (1993). *Time Series Prediction: Forecasting the Future and Understanding the Past*. Perseus Press.

WENTIAN, L. (1990) Mutual information functions versus correlation functions. *Journal of Statistical Physics*, vol. 60, nos. 5-6, pp. 823-837.

YULE G. (1927). On a method of investigating periodicities in disturbed series with special reference to Wolfer's sunspot numbers. *Philos. Trans. R. Soci.*, A226.