Recovering Clipped OFDM Symbols with Bayesian Inference and Parallel Processing

1 Background and Overview

Orthogonal frequency-division multiplexing (OFDM) is a well known and widely used method of encoding digital data on multiple carrier frequencies, and considered one of the fundamental technologies that 4G/4G-LTE Network Frame incorporates. The basic idea of OFDM is to transmit multiple streams of information by modulating them onto a set of orthogonal carrier frequency. Because the carrier frequencies are orthogonal, theoretically the Inter-Symbol Inference (ISI) of different streams at the same time is systematically cancelled. (see Figure 1)

However, OFDM system typically employs a frequency domain modulation, and thus a bounded frequency domain constellation would lead to a large excursion of the time domain signal from its average power, although the average power is the same in time domain and frequency domain according to Parseval's theorem. As a consequence, in an OFDM transmitter, the signal entering the High Power Amplifier (HPA) has a extreme high peak to average ratio (PAR), this would force the HPA to operate in its nonlinear region, which is considered one of the drawbacks of OFDM.

The behavior of HPA at its nonlinear range causes a clipping distortion on the output signal (see Figure 2). Although multiple PAR reduction methods have been introduced to the industry, the transmitted signals are still clipped with non-negligible probability, thus it is useful to develop a way for recovering clipped symbols at the receiver.

Applying Machine learning methods with Bayesian inference to communication area is recently proved fruitful in channel state estimation. This project is inspired by a paper published in 2000 [1] and dedicated to finding a new combination of Machine learning techniques and communication engineering, along with parallel data processing. The idea of the proposed algorithm is to make use of the information contained in the undistorted samples to reconstruct the values of the peaks that have been clipped.

Furthermore, the proposed algorithm is applicable to parallel processing, which is useful in reducing the DSP time consumption. The algorithm is first simulated on Matlab for the verification of correctness simulation, and then implemented on Spark [3] for the verification of parallel/distribution applicability.

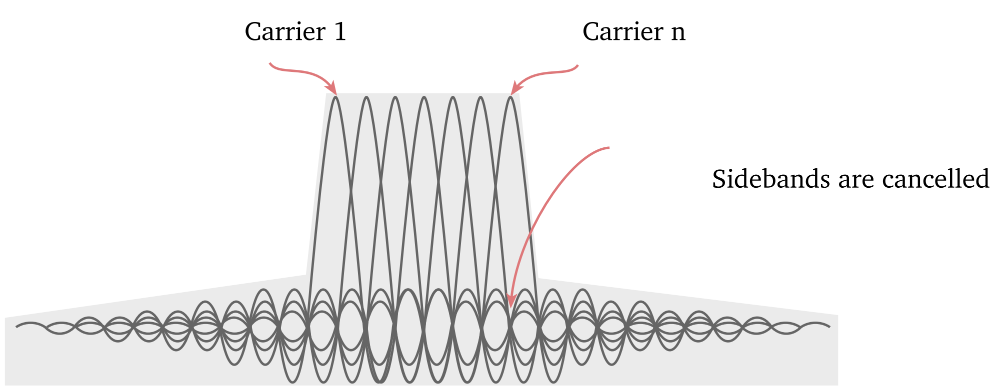

Figure 1: OFDM principle

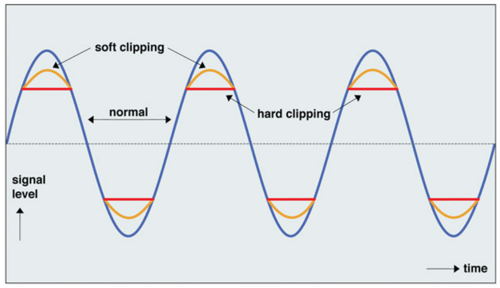

Figure 2: Clipping distortion

2 Problem Formulation

The OFDM modulation is well known and extensively studied. As a typical example of OFDM, given the frequency domain symbol $\mathbf{X} = \{X_0, \cdots, X_{N-1}\}$, the output of the modulator is given by:

$$x_n = \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} X_l e^{2j\pi \frac{nl}{N}}$$
$$X_l \in \mathcal{S}, \quad |\mathcal{S}| = Q$$

where N is the length of symbol sequence (i.e., the number of carrier frequencies), with a typical value from 128 to 2048. The output denoted by $\mathbf{x} = \{x_1, \cdots, x_{N-1}\}$ is the time domain symbol, which will be further put into an HPA. $\mathcal{S}$ is the set of constellations, and Q is the number of constellations that are used in frequency domain signal generation.
We assume a hard clipping situation, which makes the output of the HPA (i.e. the actual transmitted time domain signal) to be:

$$\tilde{x}_n = \begin{cases} x_n & , \text{if } |x_n| \le A \\ Ae^{j\text{Arg}(x_n)} & , \text{if } |x_n| > A \end{cases}$$

where A^2 is the saturation power of the amplifier, and Arg(x_n) is the phase of the complex value x_n. Besides, a quantity that measures the severity of distortion is the *Input-BackOff* (IBO) defined by $(IBO)_{dB} = 10\log_{10}(\frac{A^2}{\mathbf{E}[x_n^2]})$.
Further, we assume that the channel is time invariant frequency selective, with an impulse respond $\{h_m\}$, $m = 0, 1, \cdots, M$ (M is the maximum delay spread). Thus the received signal is given by:

$$y_n = \sum_{m=0}^{M-1} h_m \tilde{x}_{n-m} + w_n, \quad n = 0, 1, \cdots, N-1$$

where $\mathbf{w} = \{w_n\}$ denotes an Additive White Gaussian Noise. Back to the frequency domain, we have:

$$Y_l = H_l \tilde{X}_l + W_l, \quad l = 0, 1, \cdots, N-1$$

with H being the channel transfer function, and $\mathbf{W}$ being an zero-mean AWGN with the same variance as $\mathbf{w}$. Also, for simplicity, we will neglect the imperfect of synchronization and equalization, and also neglect the inference of the prefix cyclic. Then the frequency signal at the receiver after the equalization is given by:

$$\bar{Y}_l = \frac{Y_l}{H_l} = \tilde{X}_l + \frac{W_l}{H_l}$$

It is shown by a variety textbook that, the corresponding equalized time domain signal is indeed the AWGN-noised clipped signal, i.e.

$$\bar{y}_n = \tilde{x}_n + \epsilon_n \quad , \sigma_\epsilon^2 = \frac{\sigma_w^2}{N} \sum_{l=1}^{N} \frac{1}{|H_l|^2}$$

3 Bayesian Model

The objective of this project is to recover original signal ($\mathbf{x}$ or $\mathbf{X}$) out of $\bar{\mathbf{y}} = \{\bar{y}_n\}$. To do so, a Bayesian MAP approach is incorporated. Note that although we don't consider the effect of imperfect equalization, but the equivalent noise variance σ_ϵ^2 still remains a random variable.
We then introduce the indicator variable, i.e. N binary augmented random variables (hyperparameter) $\mathbf{z} = \{z_n\}$ denoting whether each symbol is clipped, which leads us to:

$$\bar{y}_n = \tilde{x}_n + \epsilon_n = x_n(1 - z_n) + Ae^{j\mathrm{Arg}(x_n)} z_n + \epsilon_n, \quad z_n \in \{0, 1\}$$

and $\mathbf{z}$ should be estimated jointly with the desired input $\mathbf{x}$.
Now we make the assumption of the prior to $\mathbf{x}$, $\mathbf{z}$ and σ_ϵ^2:
Indicator variable: Since the receiver gets no prior information of which symbol is clipped, we choose the equipropable Bernoulli distribution as the prior:

$$\Pr(z_n) = \begin{cases} 1/2 & , z_n = 0 \\ 1/2 & , z_n = 1 \end{cases} \quad , n = 0, 1, \cdots, N - 1$$

Further, if the receiver gets the information of the probability each symbol being clipped (relative to IBO), the probability in the above prior could be modified to suit the knowledge.
Information symbol: Instead of assuming a prior to the time domain signal, since the frequency domain symbols are drawn from a finite set of constellation, assuming that each constellation is equiprobable, we would have:

$$\Pr(X_l = S_u) = \frac{1}{Q}$$

Noise variance: For a variance parameter, it is computationally efficient and commonly used to choose a conjugate prior, which is, in this case, an inverse Gamma prior:

$$\Pr(\sigma_\epsilon^2) = \mathcal{IG}(a_0, b_0)$$

where a_0, b_0 are hyperparameters that could reflect the prior knowledge of the channel state.
Also, the likelihood of a given equalized symbol set is easy to get:

$$\begin{aligned} \Pr(\bar{\mathbf{y}}|\mathbf{x}, \mathbf{z}, \sigma_\epsilon^2) &= \prod_{n=1}^{N} \Pr(\bar{y}_n | x_n, z_n, \sigma_\epsilon^2) \\ &= \prod_{n: z_n = 0} \mathcal{N}(\bar{y}_n | x_n, \sigma_\epsilon^2) \times \prod_{n: z_n = 1} \mathcal{N}(\bar{y}_n | Ae^{j\mathrm{Arg}(x_n)}, \sigma_\epsilon^2) \end{aligned}$$

However, in practice, direct maximization of the full posterior $\Pr(\mathbf{x}, \mathbf{z}, \sigma_\epsilon^2 | \bar{\mathbf{y}})$ with respect to all variables at the same time is hard to implement, since the parameter space is too large. To deal with this issue, we will now just consider the posterior on each set of variable, given other sets fixed, and these *partial-posterior* will be uesd in the algorithm described in next section.
Indicator variable: It is easy to verify that, the posterior of $\mathbf{z}$ given $\mathbf{x}$ and σ_ϵ^2 is still a Bernoulli distribution, but with a different probability. Also, the posterior is independent for different n, that is, for a particular n:

$$\begin{aligned} \Pr(z_n = 0 | \bar{y}_n, x_n, \sigma_\epsilon^2) &\propto \Pr(z_n = 0) \Pr(\bar{y}_n | x_n, z_n = 0, \sigma_\epsilon^2) \\ &\propto \frac{1}{2} \mathcal{N}(\bar{y}_n | x_n, \sigma_\epsilon^2) \\ \Pr(z_n = 1 | \bar{y}_n, x_n, \sigma_\epsilon^2) &\propto \frac{1}{2} \mathcal{N}(\bar{y}_n | Ae^{j\mathrm{Arg}(x_n)}, \sigma_\epsilon^2) \end{aligned}$$

Thus, the probability of the Bernoulli distribution is:

$$
\begin{aligned}
p_{z_n} &= \frac{\Pr(z_n = 1|\bar{y}_n, x_n, \sigma_\epsilon^2)}{\Pr(z_n = 0|\bar{y}_n, x_n, \sigma_\epsilon^2) + \Pr(z_n = 1|\bar{y}_n, x_n, \sigma_\epsilon^2)} \\
&= \frac{\mathcal{N}(\bar{y}_n|Ae^{j\mathrm{Arg}(x_n)}, \sigma_\epsilon^2)}{\mathcal{N}(\bar{y}_n|x_n, \sigma_\epsilon^2) + \mathcal{N}(\bar{y}_n|Ae^{j\mathrm{Arg}(x_n)}, \sigma_\epsilon^2)} \\
&= \left(1 + \frac{\mathcal{N}(\bar{y}_n|Ae^{j\mathrm{Arg}(x_n)}, \sigma_\epsilon^2)}{\mathcal{N}(\bar{y}_n|x_n, \sigma_\epsilon^2)}\right)^{-1} \\
&= \left(1 + \exp\left\{-\frac{1}{\sigma_\epsilon^2}\left[\left|\bar{y}_n - Ae^{j\mathrm{Arg}(x_n)}\right|^2 - |\bar{y}_n - x_n|^2\right]\right\}\right)^{-1} \\
&= \mathrm{Sigmoid}(\frac{1}{\sigma_\epsilon^2}\left[\left|\bar{y}_n - Ae^{j\mathrm{Arg}(x_n)}\right|^2 - |\bar{y}_n - x_n|^2\right])
\end{aligned}
$$

Information symbol: We could compute the posterior of each element of $\mathbf{X}$ given $\mathbf{z}$, σ_ϵ^2 and other element of $\mathbf{X}$ as:

$$
\begin{aligned}
&\Pr(X_l = S_u|\mathbf{X}_{-l}, \bar{\mathbf{y}}, \mathbf{z}, \sigma_\epsilon^2) \\
&\propto \Pr(\bar{\mathbf{y}}|\mathbf{X}_{-l}, X_l = S_u, \mathbf{z}, \sigma_\epsilon^2)\Pr(X_l = S_u) \\
&\propto \prod_{n=1}^{N} \Pr(\bar{y}_n|\mathbf{X}_{-l}, X_l = S_u, z_n, \sigma_\epsilon^2)\Pr(X_l = S_u) \\
&\propto \prod_{n=1}^{N} \Pr(\bar{y}_n|x_{n,X_l=S_u}, z_n, \sigma_\epsilon^2)\frac{1}{Q} \\
&\propto \prod_{n:z_n=0} \mathcal{N}(\bar{y}_n|x_{n,X_l=S_u}, \sigma_\epsilon^2) \times \prod_{n:z_n=1} \mathcal{N}(\bar{y}_n|Ae^{j\mathrm{Arg}(x_{n,X_l=S_u})}, \sigma_\epsilon^2) \\
&\propto \exp\left\{\sum_{n:z_n=0} |\bar{y}_n - x_{n,X_l=S_u}|^2 + \sum_{n:z_n=1} \left|\bar{y}_n - Ae^{j\mathrm{Arg}(x_{n,X_l=S_u})}\right|^2\right\}
\end{aligned}
$$

Where $\mathbf{X}_{-l} = \{X_{l'}\big|l' \neq l\}$, and $x_{n,X_l=S_u}$ is the time domain symbol corresponding to setting the l-th frequency domain symbol to S_u, with

$$
x_{n,X_l=S_u} = x_n + \frac{1}{\sqrt{N}} + (S_u - X_l)e^{\frac{2j\pi nl}{N}}
$$

Information symbol: Given the information symbols $\mathbf{x}$ and the indicator variables $\mathbf{z}$, the posterior of noise variance σ_ϵ^2 is given by:

$$
\begin{aligned}
\Pr(\sigma_\epsilon^2|\bar{\mathbf{y}}, \mathbf{x}, \mathbf{z}) &\propto \Pr(\bar{\mathbf{y}}|\mathbf{x}, \mathbf{z}, \sigma_\epsilon^2)\Pr(\sigma_\epsilon^2) \\
&\propto \mathcal{IG}(a_N, b_N)
\end{aligned}
$$

where a_N and b_N is computed by:

$$
\begin{aligned}
a_N &= a_0 + \frac{N}{2} \\
b_N &= b_0 + \frac{1}{2}\sum_{n=1}^{N}\left(\bar{y}_n - x_n(1 - z_n) - Ae^{j\mathrm{Arg}(x_n)}z_n\right)^2
\end{aligned}
$$

4 Iterative Algorithm

We now introduce an iterative/stochastic MAP algorithm that deals with the problem proposed above. The algorithm updates the three set of variables one at a time sequentially, with a demand of running space much lower than maximizing the full posterior. Besides, this algorithm is parallelization capable, which would be described in next section. In fact, the algorithm could be interpreted as an extension of the *Expectation-Maximization* strategy.

Algorithm 1: Iterative/Stochastic MAP Estimation

Result: transmitted signal $\hat{\mathbf{x}}$, hyperparameter $\hat{\mathbf{z}}$, receiver noise power $\hat{\sigma}_\epsilon^2$

1 Compute $\bar{\mathbf{y}}$ at the receiver;

2 Start point $\mathbf{x}^{(0)}$, $\mathbf{z}^{(0)}$, $\sigma_\epsilon^{(0)}$ drawn from the given prior;

3 $k \leftarrow 1$;

4 **do**

5   Drawn $\mathbf{z}^{(k)} \sim \Pr(\mathbf{z}\big|\mathbf{x}^{(k-1)}, \sigma_\epsilon^{(k-1)}, \bar{\mathbf{y}})$;

6   Update: $\mathbf{x}^{(k)} = \arg\max_{\mathbf{x}} \Pr(\mathbf{x}\big|\mathbf{z}^{(k)}, \sigma_\epsilon^{(k-1)}, \bar{\mathbf{y}})$;

7   Drawn $(\sigma_\epsilon^2)^{(k)} \sim \Pr(\sigma_\epsilon^2\big|\mathbf{z}^{(k)}, \mathbf{x}^{(k)}, \bar{\mathbf{y}})$;

8     $k \leftarrow k + 1$;

9 **while** $k \leq K_{max}$;

10 **return** $\left(\hat{\mathbf{x}}, \hat{\mathbf{z}}, \hat{\sigma}_\epsilon^2\right) \leftarrow \left(\mathbf{x}^{(k)}, \mathbf{z}^{(k)}, (\sigma_\epsilon^2)^{(k)}\right)$

The basic idea of algorithm 1 is that, in each iteration, we first update the indicator variables $\mathbf{z}$ by drawing them from the Bernoulli posterior given above. Then, the frequency domain information symbols $\mathbf{X}$ are updated, by choosing the most likely constellation according to the posterior. Finally, the noise variance is updated using the result of the first two steps. Specifically, Within the step 6 that updates $\mathbf{x}$, the frequency domain symbols is get by:

$$\begin{aligned} X_l^{(k)} &= \arg\max_{S_u} \Pr(X_l^{(k)} = S_u|\mathbf{X}_{-l}^{(k-1)}, \bar{\mathbf{y}}, \mathbf{z}^{(k-1)}, (\sigma_\epsilon^2)^{(k-1)}) \\ &= \arg\max_{S_u} \sum_{n:z_n^{(k)}=0} \left|\bar{y}_n - x_{n,X_l=S_u}^{(k-1)}\right|^2 + \sum_{n:z_n^{(k-1)}=1} \left|\bar{y}_n - Ae^{j\text{Arg}(x_{n,X_l=S_u}^{(k-1)})}\right|^2 \end{aligned}$$

and the time domain signal is then computed through inverse DFT.
The convergence of Algorithm 1 is previously proved, but the stationary point is not necessary the global maximum of the full posterior density, and thus the algorithm is relatively sensitive to the initialization.

5 Implementation and Parallelization

In fact, one could easily realizes that Algorithm 1 is parallelization capable. Specifically, step 5 updating $\mathbf{z}$ could be processed parallelly on every single n, since the probability p_{z_n} doesn't depend on any other information but $\bar{\mathbf{y}}$, $\mathbf{x}_n^{(k-1)}$ at n-th position. Also, step 6 updating frequency domain $\mathbf{X}$ could also be put into parallel by $l = 0, 1, \cdots, N-1$, since the computation of $x_{n,X_l=S_u}$ is independent for each set of (n, l).
In the parallel implementation, two RDDs are maintained, one consists of the time domain variable tuple (i.e. $(x_n, \bar{y}_n, z_n)$), and the other consists of the frequency domain information symbols X_l. Within each iteration, the update of $\mathbf{x}$, $\mathbf{z}$ and σ_ϵ^2 are done by a series of map-reduce operations of the two RDDs.
In this project, Algorithm 1 is first implemented in Matlab for the verification of correctness, then implement in Apache Spark for the verification of parallelization feasibility.
In the simulation, a simple convolution channel is used with $M = 5$, the channel transfer function is shown in Figure 4. Besides, a power-normalized 16-QAM constellation is used for the frequency domain information symbols, shown in Figure 3. N is set to 2048, and totally 20 independent trails are tested, each with $IBO = 2.93$dB, $K_{max} = 20$, and SNR varies from 20dB to 30dB. As an example, the clipped transmitted signal and the received signal in frequency domain is shown in Figure 5 and Figure 6, respectively.
For result comparison, a naive heuristic method has been implemented, which simply choose the euclidean nearest constellation point to be the estimation for each symbol in frequency domain. The symbol error rate (SER) versus channel SNR is shown in Figure 7. We could see that a huge gap between the performance of two methods, and Algorithm 1 achieves a very low SER as desired.
Further, the parallelled demo of the algorithm is run with Spark on the Discover Cluster [2], with the number of partitions varies from 1 to 20. The average run time is shown in Figure 8. We could see that proper level of parallelization could improve the time performance of Algorithm 1. However, since the parallel cost in relatively high in software-based parallelization, the average run time starts increasing with more partition number, we could expect a better performance if it is implement on hardware-based parallelization, along with acceleration of *fft*.

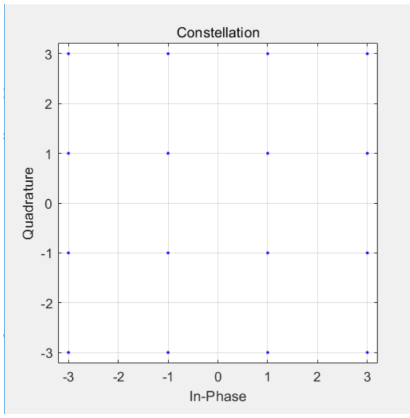

Figure 3: Frequency Domain Constellation

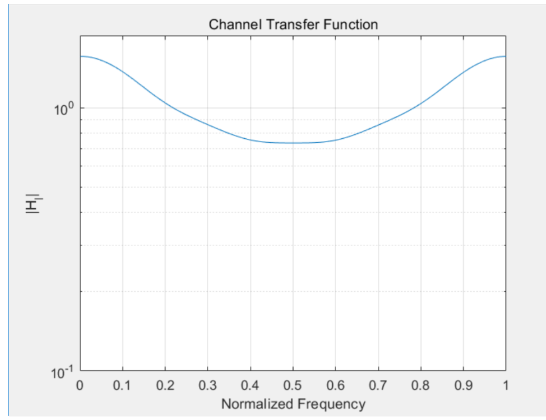

Figure 4: Channel Transfer Function

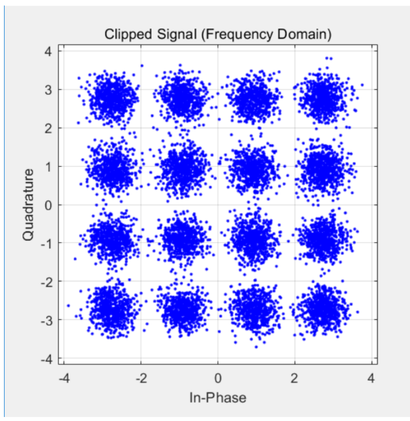

Figure 5: Clipped Transmit Signal

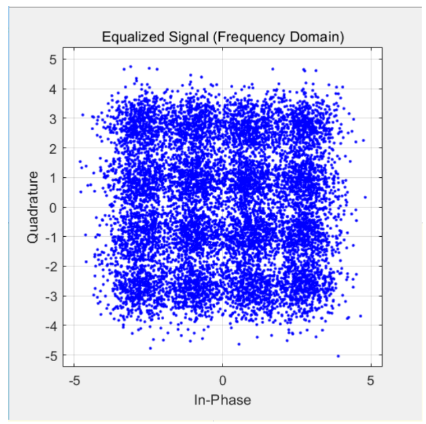

Figure 6: Equalized Signal $\bar{\mathbf{Y}}$

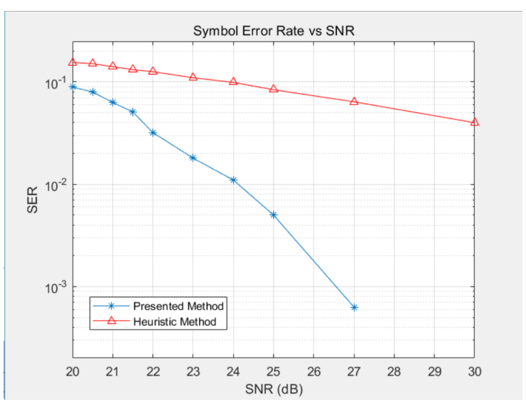

Figure 7: Symbol Error Rate

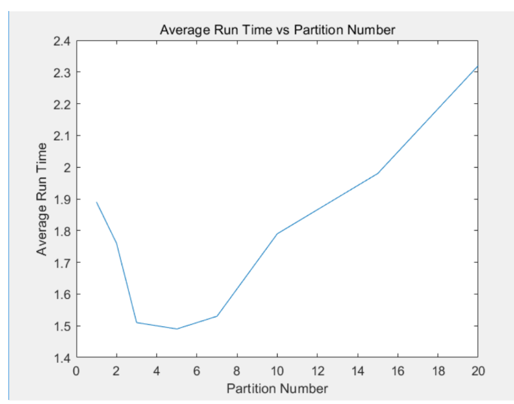

Figure 8: Parallel Running Time

References

[1] David Declercq and Georgios B Giannakis. Recovering clipped ofdm symbols with bayesian inference. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, volume 1, pages 157–160. IEEE, 2000.

[2] GitHub. Discovery cluster neu, 2019. [Online; accessed 9-Dec-2019].

[3] Wikipedia. Apache spark, 2019. [Online; accessed 9-Dec-2019].