Why Spiking Neural Networks Are Efficient: A Theorem

Michael Beer[1], Julio Urenda[2]
Olga Kosheleva[2], and Vladik Kreinovich[2]

[1]Leibniz University Hannover
30167 Hannover, Germany
beer@irz.uni-hannover.de
[2]University of Texas at El Paso
500 W. University
El Paso, Texas 79968, USA
jcurenda@utep.edu, olgak@utep.edu,
vladik@utep.edu

1. Why Spiking Neural Networks (NN)

- At this moment, artificial neural networks are the most successful – and the most promising – direction in AI.
- Artificial neural networks are largely patterned after the way the actual biological neural networks work.
- This patterning makes perfect sense:
  - after all, our brains are the result of billions of years of improving evolution,
  - so it is reasonable to conclude that many features of biological neural networks are close to optimal,
  - not very efficient features would have been filtered out in this long evolutionary process.
- However, there is an important difference between the current artificial NN and biological NN.

2. Why Spiking NN (cont-d)

- In hardware-implemented artificial NN each value is represented by the intensity of the signal.
- In contrast, in the biological neural networks, each value is represented by a frequency instantaneous spikes.
- Since simulating many other features of biological neural networks has led to many successes.
- So, a natural idea is to also try to emulate the spiking character of the biological neural networks.

3. Spiking Neural Networks Are Indeed Efficient

- Interestingly, adding spiking to artificial neural networks has indeed led to many successful applications.
- They were especially successful in processing temporal (and even spatio-temporal) signals.
- A biological explanation of the success of spiking neural networks makes perfect sense.
- However, it would be nice to supplement it with a clear mathematical explanation.
- It is especially important since:
  - in spite of all the billions years of evolution,
  - we humans are not perfect as biological beings,
  - we need medicines, surgeries, and other artificial techniques to survive, and
  - our brains often make mistakes.

4. Looking for Basic Functions

- In general, to represent a signal $x(t)$ means to approximate it as a linear combination of some basic functions.
- For example, it is reasonable to represent a periodic signal as a linear combination of sines and cosines.
- Often, it makes sense to represent the observed values as a linear combination of:
  - functions t, t^2, etc., representing the trend and
  - sines and cosines that describe the periodic part of the signal.
- We can also take into account that the amplitudes of the periodic components can also change with time.
- So, we end up with terms of the type $t \cdot \sin(\omega \cdot t)$.

5. Looking for Basic Functions (cont-d)

- For radioactivity, the observed signal is:
  - a linear combination of functions $\exp(-k \cdot t)$
  - that represent the decay of different isotopes.
- So, in precise terms, selecting a representation means selecting an appropriate family of basic functions.
- In general, elements $b(t)$ of a family can be described as $b(t) = B(c_1, \ldots, c_n, t)$ corr. to diff. $c = (c_1, \ldots, c_n)$.
- Sometimes, there is only one parameter, as in sines and cosines.
- In control, typical are functions $\exp(-k \cdot t) \cdot \sin(\omega \cdot t)$, with two parameters k and ω, etc.

6. Dependence on Parameters Is Continuous

- We want the dependence $B(c_1, \ldots, c_n, t)$ to be computable.
- It is known that all computable functions are, in some reasonable sense, continuous.
- Indeed, in real life, we can only determine the values of all physical quantities c_i with some accuracy.
- Measurements are always not 100% accurate, and computations always involve some rounding.
- For any given accuracy, we can provide the value with this accuracy.
- Thus, the approximate values of c_i are the only thing that $B(c_1, \ldots, c_n, t)$-computing algorithm can use.
- This algorithm can ask for more and more accurate values of c_i.

7. Dependence Is Continuous (cont-d)

- However, at some point it must produce the result.
- At this point, we only known approximate values of c_i.
- So, we only know the interval of possible values of c_i.
- And for all the values of c_i from this interval:
  - the result of the algorithm provides, with the given accuracy,
  - the approximation to the desired value $B(c_1, \ldots, c_n, t)$.
- This is exactly what continuity is about!
- One has to be careful here, since the real-life processes may actually be discontinuous.
- Sudden collapses, explosions, fractures do happen.

8. Dependence Is Continuous (cont-d)

- For example, we want to make sure that:
  - a step-function which is equal to 0 for $t < 0$ and to 1 for $t \geq 0$ is close to
  - an "almost" step function which is equal to 0 for $t < 0$, to 1 for $t \geq \varepsilon$ and to t/ε for $t \in (0, \varepsilon)$.
- In such situations:
  - we cannot exactly describe the value at moment t,
  - since the moment t is also measured approximately.
- What we can describe is its values at a moment close to t.

9. Dependence Is Continuous (cont-d)

- In other words, we can say that the two functions $a_1(t)$ and $a_2(t)$ are ε-close if:
  - for each t_1, there are ε-close t_{21}, t_{22} such that $a_1(t_1)$ is ε-close to a convex combination of $a_2(t_{2i})$;
  - for each t_2, there are ε-t_{11}, t_{12} such that $a_2(t_2)$ is ε-close to a convex combination of $a_1(t_{1i})$.

10. Additional Requirement

- We consider linear combinations of basic functions.
- So, it does not make sense to have two basic functions that differ only by a constant.
- If $b_2(t) = C \cdot b_1(t)$, then there is no need to consider the function $b_2(t)$ at all.
- In each linear combination we can replace $b_2(t)$ with

$$C \cdot b_1(t).$$

11. We Would Like to Have the Simplest Possible Family

- How many parameters c_i do we need? The fewer parameters:
  - the easier it is to adjust the values of these parameters, and
  - the smaller the probability of *overfitting* – a known problem of machine learning and data analysis in general.
- We cannot have a family with no parameters at all; this would mean, in effect, that:
  - we have only one basic function $b(t)$ and
  - we approximate every signal by an expression $C \cdot b(t)$ obtained by its scaling.

12. Simplest Possible Family (cont-d)

- This will be a very lousy approximation to real-life processes:
  - these processes are all different,
  - they do not resemble each other at all.
- So, we need at least one parameter.
- We are looking for the simplest possible family.
- So, we should therefore consider families depending on a single parameter c_1.
- In precise terms, we need functions $b(t) = B(c_1, t)$ corresponding to different values of the parameter c_1.

13. Most Observed Processes Are Limited in Time

- From our viewpoint, we may view astronomical processes as going on forever.
- In reality, even they are limited by billions of years.
- In general, the vast majority of processes that we observe and that we want to predict are limited in time.
- A thunderstorm stops, a hurricane ends, after-shocks of an earthquake stop, etc.
- From this viewpoint:
  - to get a reasonable description of such processes,
  - it is desirable to have basic functions which are also limited in time,
  - i.e., which are equal to 0 outside some finite time interval.

14. Limited in Time (cont-d)

- This need for finite duration is one of the main reasons in many practical problems:
  - a decomposition into wavelets performs much better than
  - a more traditional Fourier expansion into linear combinations of sines and cosines.

15. Shift- and Scale-Invariance

- Processes can start at any moment of time.
- Suppose that we have a process starting at moment 0 which is described by a function $x(t)$.
- What if we start the same process t_0 moments earlier?
- At each moment t, the new process $x'(t)$ has been happening for the time period $t + t_0$, so $x'(t) = x(t + t_0)$.
- There is no special starting point.
- So it is reasonable to require that the class of basic function not change if we change the starting point:

$$\{B(c_1, t + t_0)\}_{c_1} = \{B(c_1, t)\}_{c_1}.$$

- Similarly, processes can have different speed.

16. Shift- and Scale-Invariance (cont-d)

- Some processes are slow, some are faster:
  - if a process starting at 0 is $x(t)$,
  - then a λ times faster process is characterized by the function $x'(t) = x(\lambda \cdot t)$.
- There is no special speed.
- So it is reasonable to require that the class of basic function not change if we change the process's speed:

$$\{B(c_1, \lambda \cdot t\}_{c_1} = \{B(c_1, t)\}_{c_1}.$$

- Now, we are ready for the formal definitions.

17. Definitions and the First Result

- We say that a function $b(t)$ is *limited in time* if it equal to 0 outside some interval.
- We say that a function $b(t)$ is a *spike* if it is different from 0 only for a single value t.
- This non-zero value is called the *height* of the spike.
- Let $\varepsilon > 0$ be a real number.
- We say that the numbers a_1 and a_2 are ε-*close* if

$$|a_1 - a_2| \leq \varepsilon.$$

- We already had a definition of the functions $a_1(t)$ and $a_2(t)$ being ε-*close*.

18. Definitions and the First Result (cont-d)

- We say that a mapping $B(c_1, t)$ is *continuous* if, for every c_1 and $\varepsilon > 0$, there exists $\delta > 0$ such that:
  - if c_1' is δ-close to c_1,
  - then the function $b(t) = B(c_1, t)$ is ε-close to the function $b'(t) = B(c_1', t)$.
- By a *family of basic functions*, we mean a continuous mapping for which:
  - for each c_1, the function $b(t) = B(c_1, t)$ is limited in time, and
  - if $c_1 \neq c_1'$, then $B(c_1', t) \not\equiv C \cdot B(c_1, t)$.
- We say that a family $B(c_1, t)$ is *shift-invariant* if for each t_0: $\{B(c_1, t)\}_{c_1} = \{B(c_1, t + t_0)\}_{c_1}$.
- We say that a family $B(c_1, t)$ is *scale-invariant* if for each $\lambda > 0$: $\{B(c_1, t)\}_{c_1} = \{B(c_1, \lambda \cdot t)\}_{c_1}$.

19. The First Result (cont-d)

- **Proposition.** *If a family of basic functions* $B(c_1, t)$ *is shift- and scale-invariant, then:*
  - *for every* c_1, *the corresponding function* $b(t) = B(c_1, t)$ *is a spike, and*
  - *all these spikes have the same height.*
- This result provides a possible explanation for the efficiency of spikes.

20. Proof

- Let us assume that the family of basic functions $B(c_1, t)$ is shift- and scale-invariant.
- Let us prove that all the functions $b(t) = B(c_1, t)$ are spikes.
- First, we prove that none of the functions $B(c_1, t)$ is identically 0.
- Indeed, the zero function can be contained from any other function by multiplying by 0.
- This would violate the definition of a family of basic functions).
- Let us prove that each function from the given family is a spike.
- Indeed, each of the functions $b(t) = B(c_1, t)$ is not identically zero, i.e., it attains non-zero values for some t.

21. Proof (cont-d)

- By definition, each of these functions is limited in time.
- So, the values t for which the function $b(t)$ is non-zero are bounded by some interval.
- Thus, the values $t_- \stackrel{\text{def}}{=} \inf\{t : b(t) \neq 0\}$ and $t_+ \stackrel{\text{def}}{=} \sup\{t : b(t) \neq 0\}$ are finite, with $t_- \leq t_+$.
- Let us prove that we cannot have $t_- < t_+$.
- Indeed, in this case, the interval $[t_-, t_+]$ is non-degenerate; thus:
  - by an appropriate combination of shift and scaling,
  - we will be able to get this interval from any other non-degenerate interval $[a, b]$.
- The family is shift- and scale-invariant.
- Thus, the correspondingly re-scaled function $b'(t) = b(\lambda \cdot t + t_0)$ also belongs to the family $B(c_1, t)$.

22. Proof (cont-d)

- For this function, the corresponding values t'_- and t'_+ will coincide with a and b.
- All these functions are different – so, we will have a 2-dimensional family of functions.
- This contradicts to our assumption that the family $B(c_1, t)$ is one-dimensional.
- We cannot have $t_- < t_+$, so $t_- = t_+$, i.e., every function from our family is a spike.
- Let us prove that all the spikes have the same height.
- Indeed, let $b_1(t)$ and $b_2(t)$ be any two functions from the family.

23. Proof (cont-d)

- Both functions are spikes, so:
  - the value $b_1(t)$ is only different from 0 for some value t_1, its height is $h_1 \stackrel{\text{def}}{=} b_1(t_1)$;
  - similarly, the value $b_2(t)$ is only different from 0 for some value t_2, its height is $h_2 \stackrel{\text{def}}{=} b_2(t_2)$.
- Since the family $\mathcal{B}$ is shift-invariant, for $t_0 \stackrel{\text{def}}{=} t_1 - t_2$, the shifted function $b_1'(t) \stackrel{\text{def}}{=} b_1(t + t_0)$ is also in $\mathcal{B}$.
- The shifted function is non-zero when $t + t_0 = t_1$, i.e., when $t = t_1 - t_0 = t_2$, and it has the same height h_1.
- If $h_1 \neq h_2$, we would have $b_1'(t) = C \cdot b_2(t)$ for $C \neq 1$.
- Thus, the heights must be the same.
- The proposition is proven.

24. But Are Spiked Neurons Optimal?

- We showed that spikes naturally appear if we require reasonable properties like shift- and scale-invariance.
- This provides some justification for the spiked neural networks.
- However, the ultimate goal of neural networks is to solve practical problems.
- A practitioner is not interested in invariance or other mathematical properties.
- A practitioner wants to optimize some objective function.
- So, from the practitioner's viewpoint, the main question is: are spiked neurons optimal?

25. Different Practitioners Have Different Optimality Criteria

- In principle:
  - we can pick one such criterion (or two or three) and
  - analyze which families of basic functions are optimal with respect to these particular criterion.
- However, this will not be very convincing to a practitioner who has a different optimality criterion.
- An ideal explanation should work for *all* reasonable optimality criteria.
- To achieve this goal, let us analyze which optimality criteria can be considered reasonable.

26. What Is an Optimality Criterion: Analysis

- At first glance, the answer to this question may sound straightforward,
- We have an objective function $J(a)$ that assigns, to each alternative a, a numerical value $J(a)$
- We want to select an alternative for which the value of this function is the largest possible.
- If we are interested in minimizing losses, the value is the smallest possible.
- This formulation indeed describes many optimality criteria, but not all of them.
- Indeed, assume, for example, we are looking for the best method a for approximating functions.
- A natural criterion may be to minimize the mean squared approximation error $J(a)$ of the method a.

27. What Is an Optimality Criterion (cont-d)

- If there is only one method with the smallest possible mean squared error, then this method is selected.
- But what if there are several different methods with the same mean squared error.
- This, by the way, is often the case.
- In this case, we can use this non-uniqueness to optimize something else; e.g., we can select:
  - out of several methods with the same mean squared error,
  - the method for which the average computation time $T(a)$ is the smallest.
- The actual optimality criterion cannot be described by single objective function, it is more complex.

28. What Is an Optimality Criterion (cont-d)

- Namely, we say that a method a' is better than a method a if:
  - either $J(a) < J(a')$,
  - or $J(a) = J(a')$ and $T(a) < T(a')$.
- This additional criterion may still leave us with several equally good methods.
- We can use this non-uniqueness to optimize yet another criterion: e.g., worst-case computation time, etc.
- This criterion must enable us to decide which alternatives are better (or of the same quality).
- Let us denote this by $a \le a'$.
- Clearly, if $a \le a'$ and $a' \le a''$, then $a \le a''$, so the relation $\le$ must be transitive (a.k.a. *pre-orders*).

29. An Optimality Criterion Must Be Final

- In terms of the relation $\le$, optimal means better than (or of the same quality as) all other alternatives:

$$a \le a_{\text{opt}} \text{ for all } a.$$

- If we have several optimal alternatives, then we can use this non-uniqueness to optimize something else.
- So, the corresponding criterion is not final.
- For a *final* criterion, we should have only one optimal alternative.

30. An Optimality Criterion Must Be Invariant

- In real life, we deal with real-life processes $x(t)$, in which values of different quantities change with time t.
- The corresponding numerical values of time t depend:
  - on the starting point that we use for measuring time and
  - on the measuring unit.
- For example, 1 hour is equivalent to 60 minutes.
- Numerical values are different, but from the physical viewpoint, this is the same time interval.
- We are interested in a universal technique for processing data.

31. Criterion Must Be Invariant (cont-d)

- It is therefore reasonable to require that:
  - the relative quality of different techniques should not change
  - if we change the starting point for measuring time or a measuring unit.
- Let us describe all this in precise terms.

32. Definitions and the Main Result

- Let a set A be given; its elements will be called *alternatives*.
- By an *optimality criterion* $\le$ on the set A, we mean a transitive relation (i.e., a *pre-order*) on this set.
- An element $a_{\rm opt}$ is called *optimal* with respect to the criterion $\le$ is for all $a \in A$, we have $a \le a_{\rm opt}$.
- An optimality criterion is called *final* if there exists exactly one optimal alternative.
- For each family $B(c_1, t)$ and for each t_0, by its *shift* $T_{t_0}(B)$, we mean a family $B(c_1, t + t_0)$.
- We say that an optimality criterion on the class of all families is *shift-invariant* if
  - for every two families B and B' and for each t_0,
  - $B \le B'$ implies that $T_{t_0}(B) \le T_{t_0}(B')$.

33. Definitions and the Main Result (cont-d)

- For each family $B(c_1, t)$ and for each $\lambda > 0$, by its *scaling* $S_\lambda(B)$, we mean a family $B(c_1, \lambda \cdot t)$.
- We say that an optimality criterion on the class of families is *scale-invariant* if:
  - for every two families B and B' and for each $\lambda > 0$,
  - $B \leq B'$ implies that $S_\lambda(B) \leq S_\lambda(B')$.
- **Proposition.**
  - *Let $\leq$ be a final shift- and scale-invariant optimality criterion on the class of all families of basic f-s.*
  - *Then, all elements of the optimal family are spikes of the same height.*

34. Discussion

- Techniques based on representing signals as a linear combination of spikes are known to be very efficient.
- In different applications, efficiency mean different things: faster computations, more accurate results, etc.
- In different situations, we may have different optimality criteria.
- Our result shows that no matter what optimality criterion we use, spikes are optimal.
- This explains why spiking NN have been efficient in several different situations, with different criteria.

35. Proof

- Let us prove that the optimal family $B_{\rm opt}$ is itself shift- and scale-invariant.
- Then this result will follow from the previous Proposition.
- Indeed, let us consider any transformation T – be it shift or scaling.
- By definition of optimality, for any other family B, we have $B \le B_{\rm opt}$.
- In particular, for every B, this is true for $T^{-1}(B)$, i.e., $T^{-1}(B) \le B_{\rm opt}$.
- Here, T^{-1} denotes the inverse transformation.
- Due to invariance, $T^{-1}(B) \le B_{\rm opt}$ implies that $T(T^{-1}(B)) \le T(B_{\rm opt})$, i.e., that $B \le T(B_{\rm opt})$.

36. Proof (cont-d)

- This is true for each family B, thus the family $T(B_{\rm opt})$ is optimal.
- However, our optimality criterion is final, i.e., there is only one optimal family.
- Thus, we have $T(B_{\rm opt}) = B_{\rm opt}$.
- So, the optimal family $B_{\rm opt}$ is indeed invariant with respect to any of the shifts and scalings.
- Now, by applying the previous Proposition, we conclude the proof of this proposition.

37. Conclusions

- A usual way to process signals is to approximate each signal by a linear combinations of basic functions.
- Examples: sinusoids, wavelets, etc.
- In the last decades, a new approximation turned out to be very efficient in many practical applications.
- Namely, approximation of a signal by a linear combination of spikes.
- In this talk, we provide a possible theoretical explanation for this empirical success.

38. Conclusions (cont-d)

- Our main explanation is that:
  - for every reasonable optimality criterion on the class of all possible families of basic functions,
  - the optimal family is the family of spikes,
  - provided that the optimality criterion is scale- and shift-invariant.

39. Acknowledgments

This work was supported in part by the National Science Foundation grants:

- 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science),
- HRD-1242122 (Cyber-ShARE Center of Excellence).

The authors are greatly thankful for valuable discussions:

- to Nikola Kasabov and
- to all the participants of the 2019 IEEE Series of Symposia on Computational Intelligence (Xiamen, China).