Kaspersky Neuromorphic AI Conference 2025

Проблемы ANN-to-SNN конвертации и способы их устранения

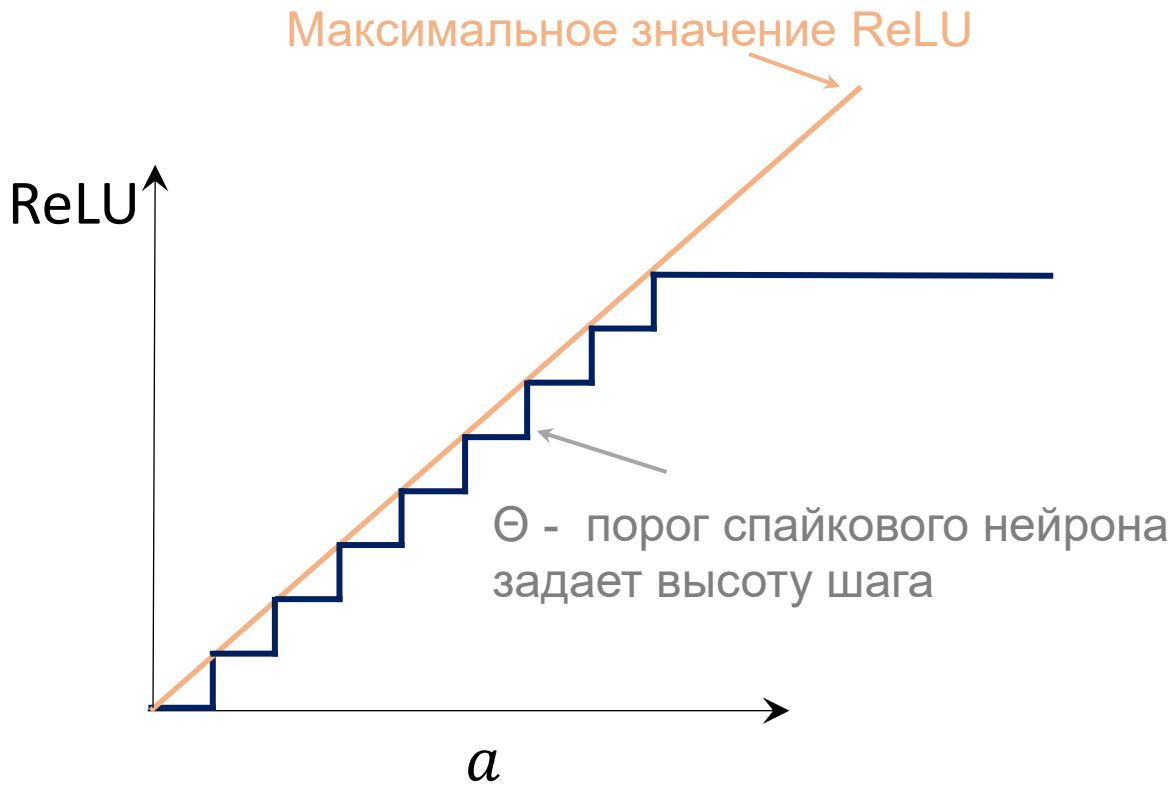
Докладчик: Рыбка Роман

НИЦ «Курчатовский институт»

Общая идея ANN-to-SNN

1) Обучение нейросетевой модели (ANN) с функцией активации ReLU;
2) Перенос весов модели на спайковую сеть (SNN) с такой же архитектурой;
3) Выбор порога спайкового нейрона для частотного способа представления информации.

ReLU соотносится с количеством спайков IF нейрона



Потенциал Integrate-and-Fire (IF) нейрона:

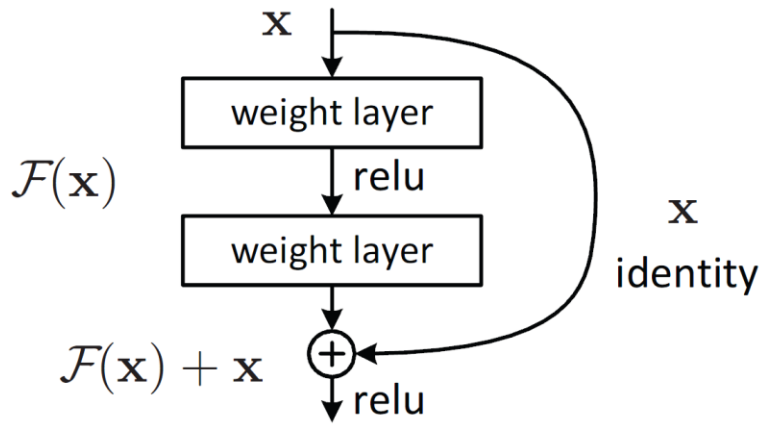
$$V(t) = V(t-1) \cdot \text{Heaviside}(\Theta - V(t-1)) + \sum_i w_i S_i(t)$$

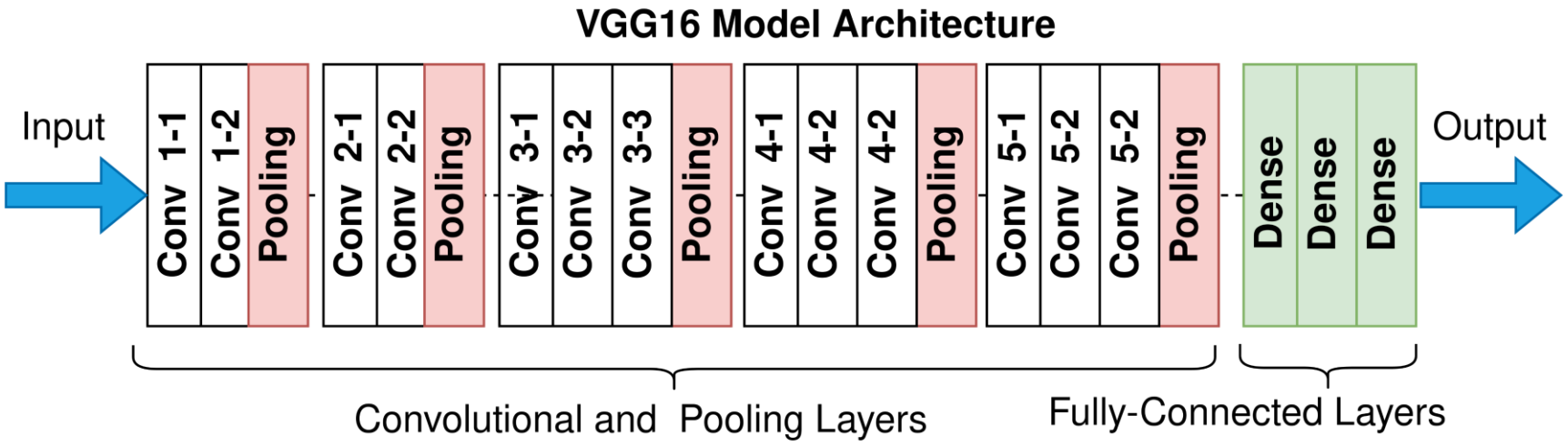
Актуальность метода

1)Простота решения: не нужно выполнять вычислительно дорогостоящее обучение спайковой сети.

2) Универсальность: подходит для предобученных ANN разных архитектур.

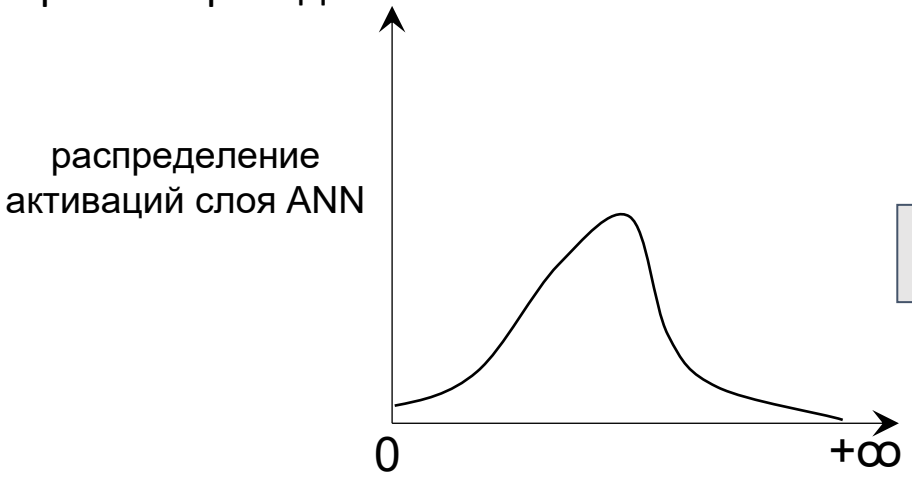
Часть ResNet сети

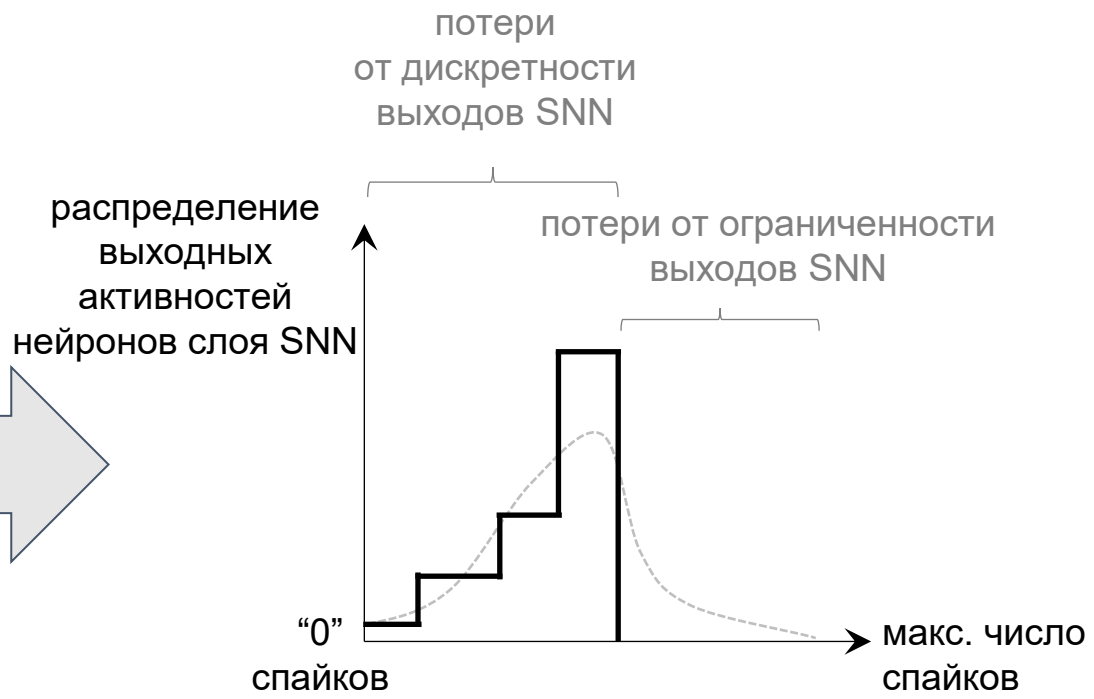




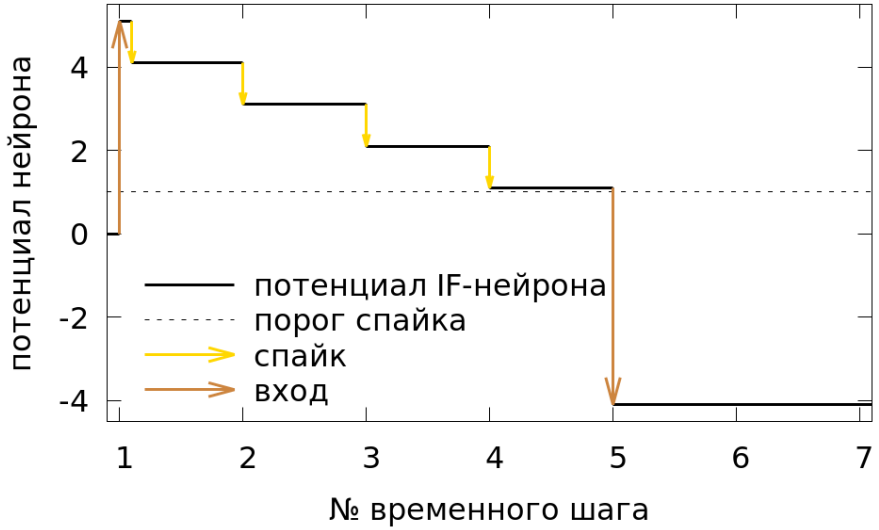
Проблемы

I. Потери от перехода к спайкам

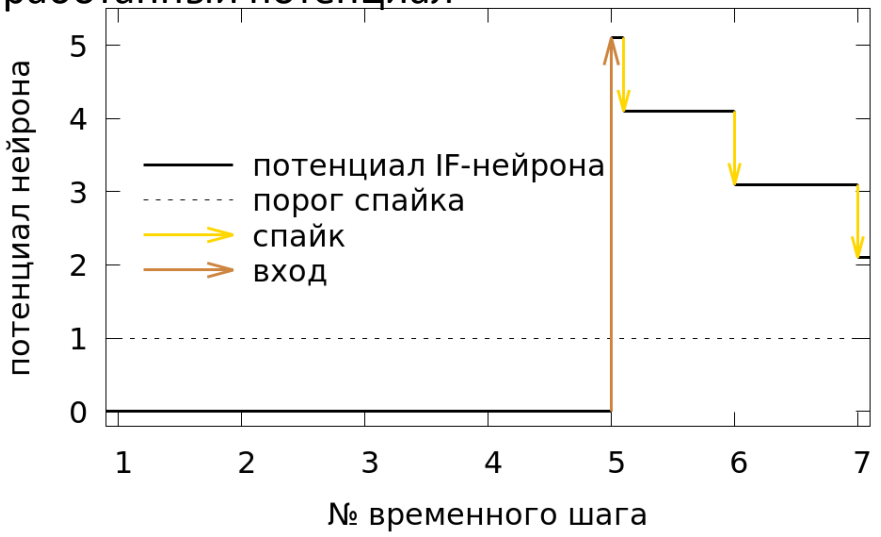




II. Авансовые спайки



III. Необработанный потенциал



1. Замена ReLU на ClipFloor

1) В (не)обученной ANN функция активации ReLU заменяется на ступенчатую: $\phi^l(T) = \text{clip}\left(\frac{\theta^l}{T}\left\lfloor\frac{a^l T}{\lambda^l}\right\rfloor, 0, \theta^l\right)$

2) (До)обучение с настройкой параметров:

θ^l, λ^l - высоты и ширины шага ступенчатой функции.

3) перенос весовых коэффициентов на связи в SNN и параметров QCFS (пороговых значений) на спайковые нейроны.

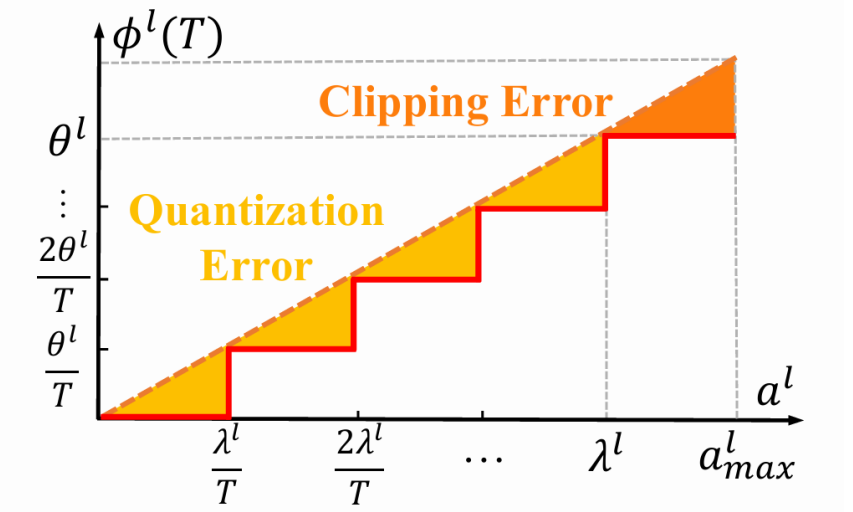


Table 2: Comparison between the proposed method and previous works on CIFAR-10 dataset.

Architecture	Method	ANN	T=2	T=8	T=32
VGG-16	RMP	93.63%	-	-	60.30%
	TSC	93.63%	-	-	-
	RNL	92.82%	-	-	85.40%
	SNNC-AP	95.72%	-	-	93.71%
	Ours	95.52%	91.18%	94.95%	95.54%
ResNet-20	RMP	91.47%	-	-	-
	TSC	91.47%	-	-	-
	Ours	91.77%	73.20%	89.55%	92.24%
ResNet-18	RTS [1]	95.46%	-	-	84.06%
	SNNC-AP [1]	95.46%	-	-	94.78%
	Ours	96.04%	75.44%	94.82%	96.08%

Что дает: избавляет от необходимости выбор порога и уменьшает ошибки от перехода к спайкам.

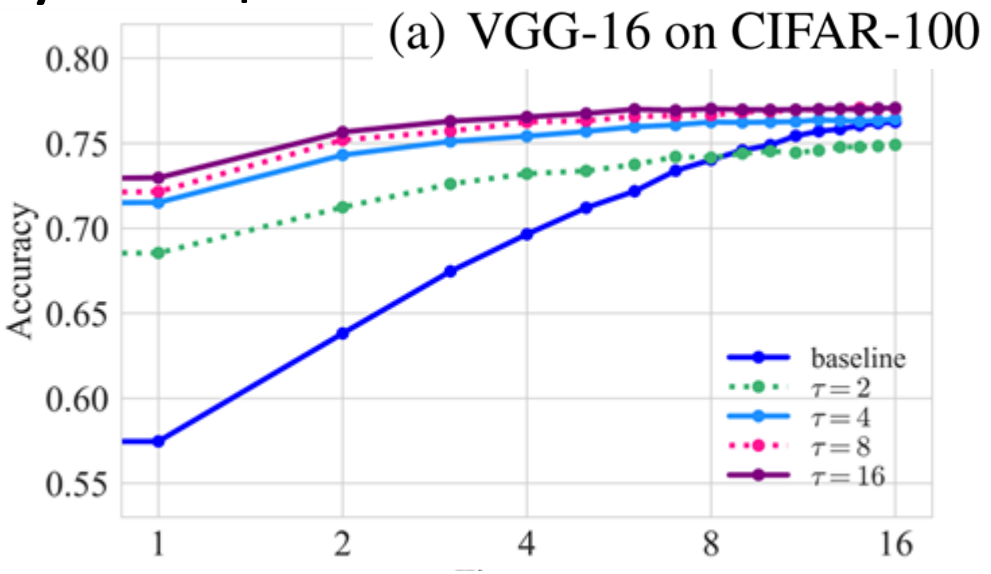
Tong Bu et al. Optimal ANN-SNN Conversion for High-accuracy and Ultra-low-latency Spiking Neural Networks

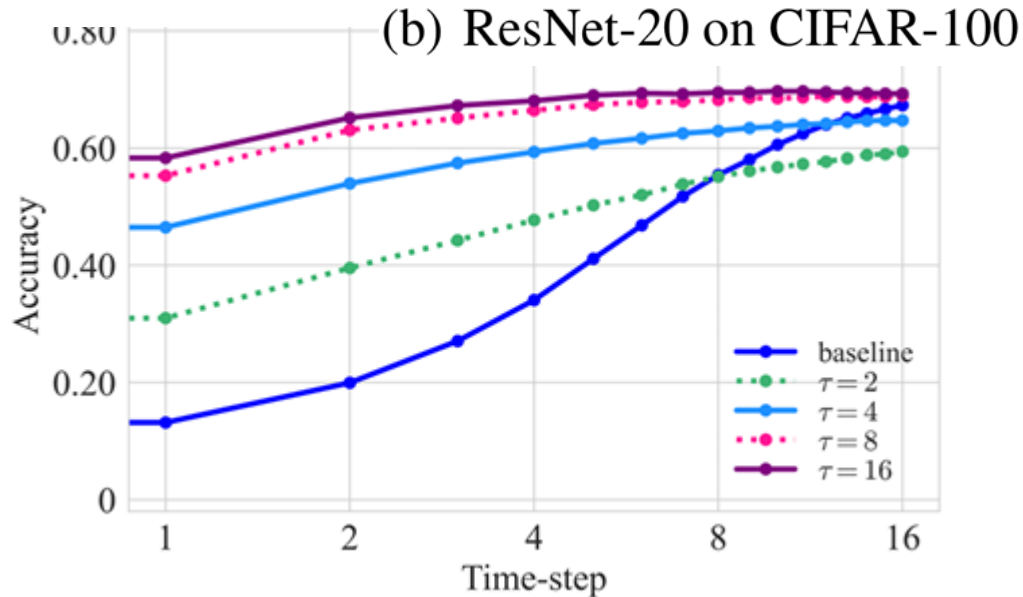
2. Продление времени симуляции

Идея: дать возможность отработать накопленный потенциал за дополнительное время симуляции.

Если за период дополнительного времени потенциал нейрона ниже порогового, нейрон «выключается».

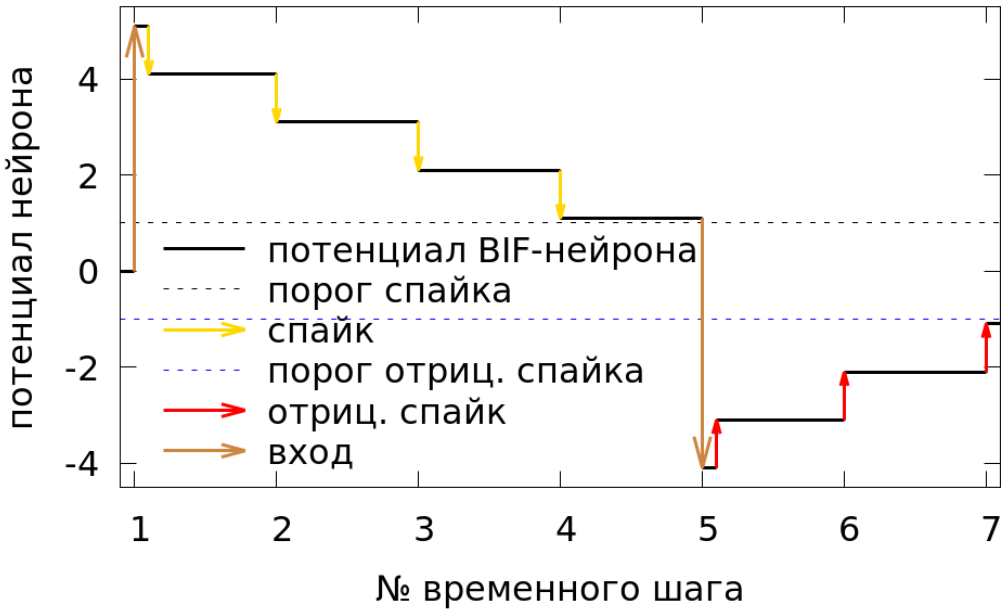
Что дает: уменьшает ошибку остаточного потенциала





Z. Hao, et al. (2023) Reducing ANN-SNN conversion error through residual membrane potential

3. Введение отрицательных спайков



mBIF нейрон:

$$\tilde{v}_j^l(t) = v_j^l(t-1) + \sum_i w_{ij} s_i^{l-1}(t) + b_j,\ \tilde{m}_j^l(t) = m_j^l(t-1),$$

$$s_j^l(t) = \begin{cases} \theta_j^l, & \tilde{v}_j^l(t) \geq \theta_j^l \\ -\theta_j^l, & \tilde{v}_j^l(t) \leq -\theta_j^l \quad \text{and} \quad \tilde{m}_j^l(t) > 0 \\ 0, & \text{otherwise} \end{cases},$$

$$v_j^l(t) = \tilde{v}_j^l(t) - s_j^l(t), \quad m_j^l(t) = \tilde{m}_j^l(t) + s_j^l(t),$$

Method	Net. Arch.	ANN Acc.	T=32	T≥512
	CIFAR10			
RMP [Han *et al.*, 2020]	VGG16	93.63	60.3	93.63(T=2048)
RateNorm [Ding *et al.*, 2021]	VGG16	92.9	85.4	92.86(Unknown)
Opt. [Deng and Gu, 2021]	VGG16	92.09	92.29	92.03(T=512)
Ours (SNM+NeuronNorm)	VGG16	94.09	**93.43**	**94.1**(T=512)
RateNorm [Ding *et al.*, 2021]	ResNet18	93.06	83.95	93.45(Unknown)
Opt. [Deng and Gu, 2021]	ResNet20	92.32	93.3	93.58(T≤600)
Ours (SNM+NeuronNorm)	ResNet18	95.39	**94.03**	**95.44**(T=1024)

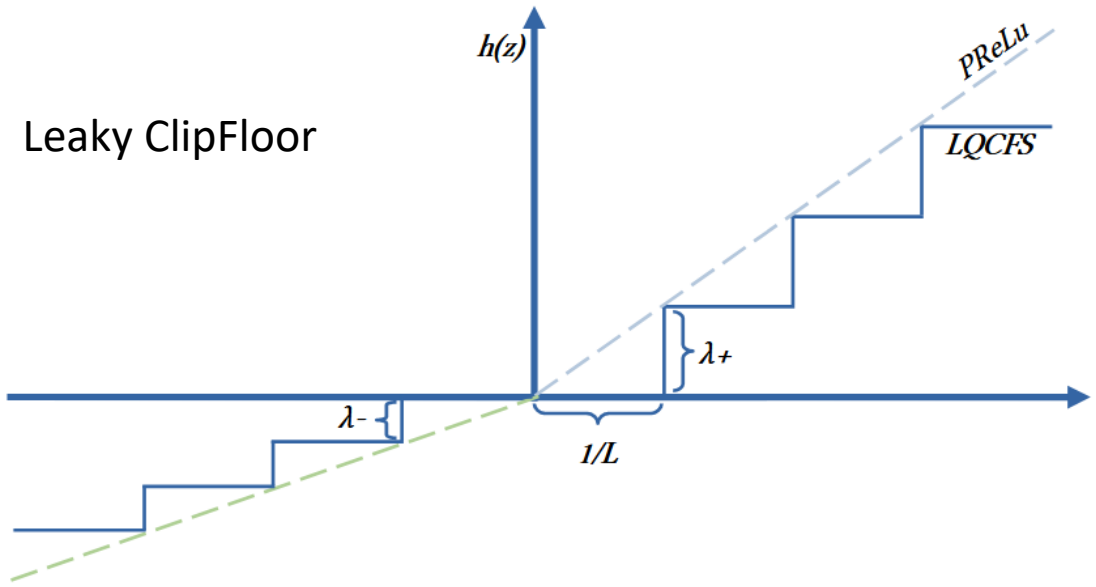
Что дает: частично решает проблему авансового спайкования

Yuchen Wang, et al. (2022, 2024) Signed Neuron with Memory: Towards Simple, Accurate and High-Efficient ANN-SNN Conversion

4. Замена на Leaky ClipFloor

В ANN применяется LeakyClipFloor как аналог функции PReLU.

В SNN используется BIF нейрон (без памяти).



$$\bar{h}(\boldsymbol{z}^l) = \begin{cases} \underset{\text{floor}}{\text{clip}}\left(\frac{1}{L}\left\lfloor\frac{\boldsymbol{z}^l L}{\lambda^l_{\text{pos}}}\right\rfloor, 0, 1\right) \\ \underset{\text{ceil}}{\text{clip}}\left(\frac{1}{L}\left\lfloor\frac{\boldsymbol{z}^l L}{\lambda^l_{\text{neg}}}\right\rfloor, -1, 0\right) \end{cases}$$

Что дает: решает проблему авансового спайкования. Увеличивает диапазон значений (за счет отрицательной части).

Сравнение по проблемам ANN-to-SNN

Ф-ии активации ANN	IF				BIF				mBIF			
	Ошибка дискретизации	Остат. Потенц. (+)	Остат. Потенц. (-)	Аванс. спайки	Ошибка дискретизации	Остат. Потенц. (+)	Остат. Потенц. (-)	Аванс. спайки	Ошибка дискретизации	Остат. Потенц. (+)	Остат. Потенц. (-)	Аванс. спайки
ReLU	+	+	+	+					+	+	+	+
PReLU					+	+	+	-				
CF	-	+	+	+					-	+	-	-
LCF					-	+	+	-				

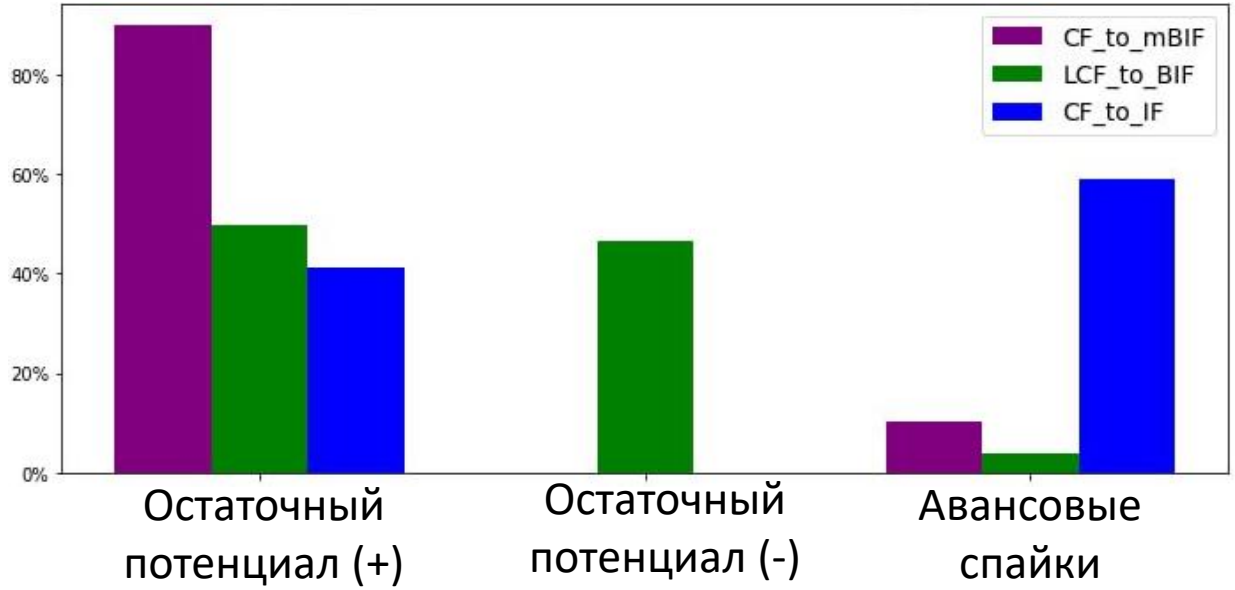
Важно: разные проблемы неодинаково влияют на итоговые результаты

Проверка в одних условиях

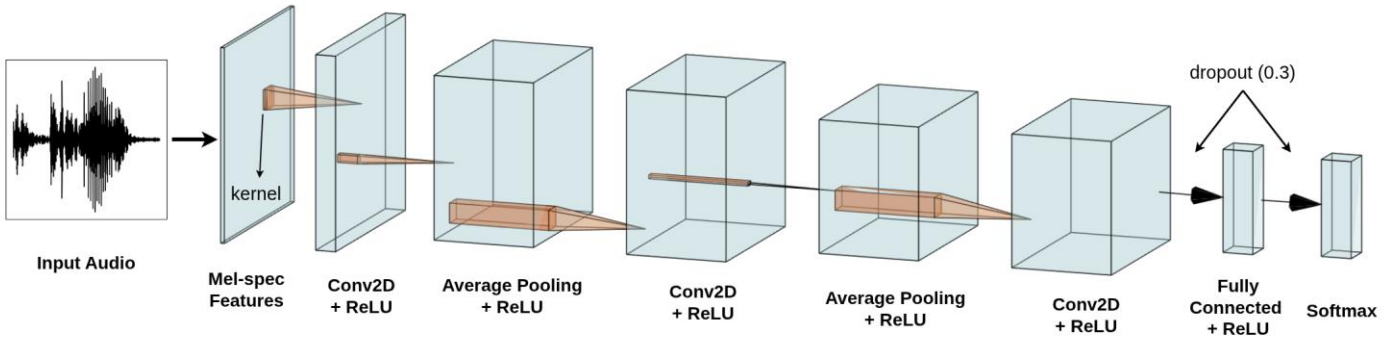
Набор данных

Набор данных **HD** содержит разговорные цифры от 0 до 9 на английском и немецком языках (20 классов). Обучающий и тестовый наборы содержат 8332 и 2088 примеров соответственно.

Доля от общего числа ошибок



Архитектура сети



Тип переноса	Точность (Accuracy)
CF -> IF	79%
CF -> mBIF	87%
LCF -> BIF	90%

Вывод и дальнейшая работа

1) ANN-to-SNN методы активно развиваются,
2) предложены несколько эффективных способов для снижения зависимости от негативных эффектов переноса.

Что нехорошо:

1) Валидация подходов осуществляется с пропуском спайковых активаций между слоями (Average Pooling),

2) Применение исходных (неспайковых) входных данных существенно повышает точность.

Спасибо за внимание!