(Sec 1)

* *What is the problem your work attacks? Be specific.*
* *Why is it an important problem?*

**Paragraph 1:**

* Introduce energy harvesting, ambient energy is weak and highly unpredictable, so devices need to be intermittently operated.
* Consequently, Deep Neural Network inference on such tiny devices has to be accumulatively executed across power cycles.
* Advanced models like transformers are increasingly popular
* Therefore, *intermittent DNN inference* is emerging as a challenging topic, particularly for advanced models, due to difficult progress preservation for complex (another term? Maybe too early to say non-deterministic) operations.
* *What is the state of the art in this topic today (if any)?*
* *What are its limits?*

**Paragraph 2 –**

* ~~Model optimizations: compression, NAS~~
  + ~~Orthogonal~~
  + ~~Optimize the model structure, not computation operations~~
* (Similar to Stateful paper, general vs. specialized approaches for intermittent inference)
* Generic intermittent inference approaches: Checkpointing (FLEX), Task-based
  + Granularity vs. overhead: intermittency management overhead grows with smaller granularity
  + Larger granularity => high re-execution costs
  + General intermittent execution incurs high runtime overheads
* Specialized: Footprinting (HAWAII/JAPARI/Stateful)
  + each job output paired with a *progress indicator*
  + parallel computation & preservation
  + Fine-grained, low re-execution, low
  + Footprinting allows much lower preservation overhead than Task-based
  + Footprinting can only be applied on deterministic (unidirectional? Mentioned in HAWAII) operations – required for correct resumption, to determine which job to resume
  + Deterministic execution flow is required to derive loop indices from the latest progress indicator (need to adapt Figure 2 in DynBal for Transformers)
  + Non-deterministic can be converted to deterministic

(notes about intermittent control-flow)

Many from Brandon Lucia; handling inconsistency – possibly different control flow after power resumption

* Intermittent computing: Challenges and opportunities
* Chain
* I/O dependent idempotence bugs in intermittent systems

(Sec 3)

* *What are the one or two key new insights in this paper?*
  + Transformers involve non-deterministic execution flow, different branches taken with different inputs
  + Why non-deterministic operations (ex: Softmax) in the middle for Tranformers? Maybe a fundamental reason (algorithmic one), check AI papers
  + During recovery, we can know the tile index from, but not the taken branch
* *How does it advance the state of the art?*
* *What makes it more effective than past approaches?*

(Why we need branch awareness? What it accomplishes?)

(Sec 4)

* *What are the key artifacts presented in your paper: a methodology, a hardware design, a software algorithm, an optimization or control technique, etc.?*
* *How were your artifacts implemented and evaluated?*

Preserving for non-determinism?

Needs shorten:

“Progress indicators for deterministic operations”

“Progress indicators for non-deterministic operations”

“Determinism-agnostic progress indicators”

Tracing? Inspired by the name in Just-In-Time compilation

Same level as footprinting?

Footprinting: tracks progress, only one control flow

Tracing: track progress for all control flows

Models with non-determinism

Transformers

RNN?

Design targets models with non-determinism, experiments focus only on Transformers

Evaluation

~~Generic baseline: Task-based (SONIC-like safe adaption)~~

Specific baseline: Only Stateful (not HAWAII)

Forced large granularity, make it deterministic

Small granularity, low accuracy (first or random branch)

Different Transformers