Product & marketing brief — improving sealed detection

Status: shareable brief as of 2026-05-18.

Audience: product, marketing, go-to-market, founders.

PDF: English PDF version

Short message

SKN already has a first lab scanner for Pokémon sealed products. It can call a dedicated endpoint, compare a photo against a sealed reference index, and return candidates with metadata. That proves the technical direction.

What is missing for reliable real-world detection is not "a better threshold". The missing product blocks are:

1. an unbiased field-evaluation dataset, separate from reference images;
2. a legally usable sealed reference catalog;
3. typology detection and packaging-specific crop;
4. OCR, metadata reranking, and user feedback.

The current accuracy is normal for a lab. To reach a shippable experience, sealed scanning must become a measurable product loop: capture, label, evaluate, correct, and enrich.

What exists today

The backend already exposes a dedicated sealed path:

- `POST /scan-sealed-lab`;
- separate `sealed_products`, `sealed_product_images`, and `sealed_product_embeddings_lab` tables;
- full-image DINOv2 visual embeddings;
- pgvector search over sealed references;
- optional filters by product type and language;
- top-K diagnostics, runner-up, score, and ambiguity reason;
- sealed mobile mode wired in the test app;
- Scan Debug screen for candidate review.

The lab uses two source types:

- `ptcg-assets` to prove the concept and reuse a broad packaging taxonomy;
- `skn-owned` references from SKN-owned photos, usable in private debug.

This foundation is useful, but it is not yet a market-accuracy claim. Clean catalog images and real phone captures are different problems.

Why sealed products are harder than cards

Raw cards have a stable format. Sealed products do not.

Main difficulties:

- shapes vary: booster, blister, display, ETB, tin, collection box, deck, bundle;
- packaging adds plastic, glare, damaged corners, folds, and varied backgrounds;
- two products can share the same artwork while differing by language, SKU, or format;
- user photos are often angled, partial, screen-captured, or poorly lit;
- public reference images are often clean packshots, not field photos;
- image rights limit what can be redistributed or displayed publicly;
- product-type confusion can send matching to the wrong family, for example booster instead of blister.

Product implication: the right UX should not only say "found / not found". It should support "likely candidate", "ambiguous", "needs confirmation", and "choose from these candidates".

What is missing to improve detection

1. Unbiased field dataset

`sealed-eval` needs real photos that are never indexed as references. Without that separation, a high score can simply prove that the system recognizes an image it has already seen.

Useful minimum targets:

Level	Target volume	Use
Useful POC	30 to 50 real photos per priority class, about 200 to 350 photos	Understand where the scanner really fails
Stable beta	100 to 200 photos per class, about 700 to 1,400 photos	Measure by typology and language
Product-grade	300+ photos per class, 2,000+ photos over time	Cover difficult cases and regression

Priority classes:

- booster pack;
- blister;
- booster display;
- Elite Trainer Box;
- collection box;
- tin / mini-tin;
- slab;
- raw card as a negative or fallback.

Each photo needs at least:

- expected product name;
- product type;
- language / region when known;
- source and rights: `owned_by_skn` or `explicit_permission`;
- capture source: real phone, screen photo, product in hand, etc.;
- status: unlabeled inbox or labelled `evaluationOnly: true`.

2. Production-safe reference catalog

`ptcg-assets` is useful for the internal lab, but it is not production-safe by itself. The repository has no declared license and likely contains protected packaging artwork.

Production needs at least one of:

- written permission from an asset source;
- a commercial provider with compatible terms;
- SKN-owned images;
- user-uploaded images with clear consent;
- a display policy that does not return candidate images when image rights are not approved.

Without a strong reference catalog, the model can be good and still fail because the correct product does not exist in the index.

3. Typology detection and sealed crop

Today, the lab often embeds the whole uploaded image. That works for a first test, but it is not optimal for real phone scans.

The first shipped improvement that does not require new training images is `preprocess=sealed`: an OpenCV foreground crop before DINOv2. It reduces table, hand, shelf, and screen noise in phone captures while keeping the original packaging aspect ratio. Early phone tests after deployment show a clear qualitative precision improvement because the visual search receives

a cleaner product signal and less background noise. This is still not a trained detector; the actual YOLO or segmentation step still needs labelled detection data and weights, and the improvement should be quantified on the separate `sealed-eval` field benchmark before it becomes a market claim.

We need a detector that can locate broad families:

- `card`;
- `slab`;
- `sealed`;
- `graded_label`;
- `barcode`;
- later: `booster_pack`, `blister`, `booster_display`, `elite_trainer_box`, `collection_box`, `tin`, `booster_bundle`.

The role of YOLO or an equivalent detector is not to identify "Base Set Blastoise Booster FR". Its role is to find the useful zone and typology. Exact identity remains the job of visual retrieval, OCR, metadata, and reranking.

Product impact:

- fewer false positives between families;
- faster scans because search can stay inside a smaller family;
- better camera UX with a frame that matches packaging, not only cards.

4. Targeted OCR and hybrid reranking

Whole-image OCR is fragile on shiny packaging. It should happen after visual retrieval.

Target pipeline:

```
user photo
-> typology detection / crop
-> visual top-K with DINOv2 + pgvector
-> targeted OCR on title, language, product code, barcode, slab label
-> hybrid reranking
-> result, ambiguity, or user choice
```

OCR is especially useful to:

- distinguish variants with the same artwork;
- separate FR / EN / DE / JP;
- read product names, set codes, or barcodes;
- later improve slabs and graded labels.

5. Feedback loop in the app

The scanner should become a product learning tool.

For each sealed scan, the app should eventually capture:

- accepted result;
- corrected result;
- "none of these";
- candidate selected from top-K;
- photo and context if the user consents to improvement use.

That lets the team prioritize improvements by real volume instead of guessing.

6. Product-readable metrics

The benchmark should be read with simple metrics:

Metric	What it tells us
Top-1 exact accuracy	Is the correct product first?
Top-K recovery	Is the correct product at least among the candidates?
Ambiguous-neighbor rate	Is the system hesitating too often?
Wrong-product-type rate	Is the scanner comparing against the wrong family?
Wrong-language rate	Are language/SKU variants causing issues?
False positive rate	Does the system assert bad matches too quickly?
P50/P95 latency	Is the scan fast enough on mobile?
Correction success	Can the user save an imperfect scan?

The new `summary.diagnoses` and `summary.nextActions` benchmark fields are designed to group failures into OCR/reranking, typology/crop, language/reference coverage, or missing-reference work.

What product and marketing can do now

Field collection

Immediate priority: build a real photo set whose rights are owned or explicitly allowed.

Use the sealed field photo collection format for the batch structure, CSV columns, naming rules, and rights metadata.

Good photos:

- full product visible;
- multiple angles;
- natural light and difficult light;
- plastic glare;
- table background, hand, shelf;
- slightly bad but realistic photo;
- screen photo if that is a likely user case.

Avoid in the test set:

- only clean packshots;
- only perfect photos;
- reusing a reference image as an evaluation image;
- mixing images without rights evidence.

Positioning

Healthy language today:

- "sealed scanner lab";
- "assisted detection with candidates and confirmation";
- "sealed-first portfolio";
- "continuous improvement through field benchmark".

Avoid until beta is measured:

- "100% accurate";
- "recognizes every Pokémon sealed product";
- "perfect pricing database";
- "guaranteed automatic identification".

Decisions to make

Product and marketing can help decide:

1. Which sealed types matter first for the demo: boosters, ETBs, displays, blisters?
2. Which language matters first: FR first, EN next, or international from day one?
3. Can we secure rights or partnerships for reference images?
4. What confidence level is acceptable before public display?
5. Should we embrace a confirmation UX instead of a single forced result at first?

Recommended roadmap

Phase	Goal	Expected result
1. Field dataset	200 to 350 labelled real photos	Know where recognition fails
2. UI typology filters	User or app indicates booster / ETB / blister	Fewer wrong families
3. Safe FR/EN references	Add owned or licensed references	Fewer missing products
4. Crop detector	Locate sealed / slab / card and crop	Better visual signal
5. OCR reranking	Read useful text over top-K	Fewer ambiguities
6. App feedback	Correct and learn from real scans	Continuous improvement

Conclusion

Sealed detection can become a real SKN differentiator because few apps treat sealed products as first-class objects. But the advantage will not come from one magic model. It will come from a complete system: clean references, field photos, typology, OCR, metrics, user correction, and clear legal rules.

The next best investment is labelled collection and honest evaluation, not premature marketing promises.

Related pages

- Sealed scanner lab
- Recognition optimization strategy
- Asset licensing and source risk register
- Scan UI/UX competitive review